

Mobile Multimedia Services in the Cloud

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der
Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung
des akademischen Grades eines Doktors der Naturwissenschaften
genehmigte Dissertation

vorgelegt von

**Diplom-Ingenieur
Dejan Kovachev**

aus Strumica, Mazedonien

Berichter: Universitätsprofessor Dr. rer. pol. Matthias Jarke
Universitätsprofessor Dr. Ing. Juan Quemada Vives
PD Dr. rer. nat. Ralf Klamma

Tag der mündlichen Prüfung: 14. Mai 2014

Diese Dissertation ist auf den Internetseiten der Hochschulbibliothek online verfügbar.

Abstract

Cloud computing is a paradigm shift promising a utility-based delivery of storage and processing capacity, services, and software over the Internet. In essence it aims to reduce costs, facilitate self-automated systems and decouple service delivery from underlying technology. Thus, the cloud paradigm empowers customers with the ability to focus on creating novel services alleviating the burden of software and hardware resource provisioning. The success of cloud computing in the domain of enterprise applications has sparked increasing interest in applying the same principles to the provision of mobile multimedia services. However, the potential benefits are far from being achieved, despite the rapid growth in popularity and omnipresence of mobile multimedia applications.

The combination of cloud computing and mobile multimedia is non-trivial, and many aspects from system, mobile multimedia and user perspectives need to be considered. For example, mobile applications in the cloud involve a trade-off in terms of what should run on the device and what in the cloud, which is contingent to the application type, the device capability, data locality and the operating environment (network bandwidth, delay, cloud availability). Moreover, the traditional server/client programming models fail to provide seamless cloud execution in volatile mobile networks. Furthermore, distant cloud data centers induce prohibitive latency for certain classes of interactive mobile applications such as 3D games and augmented reality.

This dissertation investigates ways to efficiently apply the concepts of the emerging cloud computing paradigm in the design, development and delivery of mobile multimedia services. It describes an information systems architecture called CAELUS (Cloud Architecture for Enabling Mobile Multimedia Services) which includes both conceptual models and a concrete software platform. The conceptual models capture specific requirements for efficient building of mobile multimedia cloud services and guide the creation of the software platform (i5Cloud) which serves as a test bed of the CAELUS architecture. The contributions of this dissertation, in addition to a comprehensive survey of literature, comprise a design view, platform and abstraction levels that lower the barrier for mobile multimedia services to leverage the clouds.

Several case studies have evaluated the CAELUS-based development and delivery of mobile multimedia cloud services. In particular, the case studies were conveyed in the application domains of technology-enhanced learning, digital documentation in cultural heritage and human-computer interaction. Prototype applications together with technical evaluations and user studies demonstrate the validity and applicability of the architecture and the conceptual approach.

Kurzfassung

Cloud Computing ist ein Paradigmenwechsel zur Bereitstellung von Speicher- und Verarbeitungskapazitäten, Diensten und Software über das Internet. Im Wesentlichen wird dadurch eine Kostenreduzierung, sowie die Entkopplung von Servicebereitstellung und zugrundeliegender Technologie erreicht. Somit befähigt Cloud Computing Kunden dazu, sich vollkommen auf die Entwicklung neuartiger Dienste zu konzentrieren, ohne sich um die Bereitstellung benötigter Software- und Hardware-Ressourcen kümmern zu müssen. Der Erfolg des Cloud Computing in Unternehmensanwendungen hat ein gesteigertes Interesse daran ausgelöst, die gleichen Prinzipien bei der Bereitstellung von mobile Multimedia-Diensten anzuwenden. Jedoch konnten die erwarteten Vorteile trotz des schnellen Wachstums an Popularität und Allgegenwärtigkeit von Multimedia-Applikationen bisher noch lange nicht ausgeschöpft werden.

Die Kombination von Cloud Computing und Multimedia-Diensten ist nicht trivial und erfordert die Betrachtung vieler Aspekte aus den Perspektiven des Gesamtsystems, des Nutzers und mobiler Medien. Beispielsweise wägen mobile Cloudanwendungen ab, was auf dem Gerät und was in der Cloud ausgeführt werden soll. Diese Entscheidung hängt von der Art der Applikation, der Geräteleistung, der Datenlokalität und der Betriebsumgebung (Netzwerk-Bandbreite, Verzögerung, Cloud Verfügbarkeit) ab. Darüber hinaus scheitert das traditionelle Client/Servermodell an nahtloser Cloud-Ausführung in unbeständigen Mobilfunknetzen. Außerdem verursachen entfernte Cloud-Rechenzentren Latenzen, die für bestimmte Klassen von interaktiven mobilen Anwendungen wie 3D-Spiele und Augmented Reality impraktikabel sind.

Die Dissertation eruiert Methoden zur effizienten Anwendung von Konzepten des aufstrebenden Cloud Computing Paradigmas auf Design, Entwicklung und Bereitstellung mobiler Multimedia-Dienste. Sie beschreibt eine Informationssystemarchitektur namens CAELUS (Cloud Architecture for Enabling Mobile Multimedia Services), die sowohl konzeptuelle Modelle als auch eine konkrete Softwareplattform in sich vereint. Die konzeptuellen Modelle erfassen spezifische Anforderungen für den effizienten Aufbau mobiler multimedialer Cloud-Dienste und dienen als Richtlinien zur Erstellung der Softwareplattform (i5Cloud), die das Testbett für die CAELUS Architektur darstellt. Die Beiträge dieser Dissertation beinhalten neben einem umfassenden Literaturüberblick eine Designsicht, eine Plattform sowie geeignete Abstraktionsschichten, die durch mobile Multimedia-Dienste Hürden auf dem Weg zur vorteilhaften Nutzung des Cloud Computing überwinden.

Die CAELUS-basierte Entwicklung und Bereitstellung mobiler Multimedia-Dienste wurde in mehreren Fallstudien evaluiert. Besonders wurden die Fallstudien in den Anwendungsdomänen von technologiegestütztem Lernen, digitale Dokumentation des kulturellen Erbes

und der Mensch-Computer-Interaktion durchgeführt. Prototypische Anwendungen inklusive begleitender technischer Evaluierungen und Nutzerstudien belegen die Validität und Anwendbarkeit der Architektur und ihres konzeptuellen Ansatzes.

Acknowledgments

I am most grateful towards each and every person who has taken an interest in this work, be it brief or lasting throughout the process.

First of all, I wish to express my deep gratitude to Prof. Matthias Jarke for his guidance and advice during my years at the Chair of Information Systems and Databases, RWTH Aachen University. His experience and insight into research and academics have supported me in delivering this dissertation. I was lucky to enjoy the affiliations with the Chair, B-IT Research School and the UMIC excellence cluster.

At the same time, I am highly indebted to Dr. Ralf Klamma for recommending me to pursue an academic career and for giving me the unique opportunity to work within his team. He constantly advised, encouraged and inspired my research and most importantly became a close friend. Our philosophical discussions often extended my view on the world. I feel extremely lucky to have worked under his guidance and I am thankful for all the confidence he had in me, and for the freedom, responsibilities and unselfish generosity I received. In addition, I like to thank Prof. Quemada-Vives from Spain for serving on the dissertation committee.

I am particularly grateful to my math teacher Gjorgji Serafimov. What my elementary teacher Mimoza Serafimova with arduous efforts initiated in me regarding learning, Gjorgji Serafimov pick it up and molded it into a desire and skills for the beautiful world of mathematics – things that have delineated my life.

Over the last years, it was a pleasure to have colleagues around who made life at the office more fun and with whom I worked together in one way or the other – thanks to Georgius Toubekis, Istvan Koren, Dr. Michael Derntl, Dr. Milos Kravcik, Dr. Khaled Rashed, Zinayida Petrushyna, Anna Hannemann et al. Besides the above, special thanks are to my colleagues who have also been close friends. The fruitful research collaboration with Dr. Yiwei Cao extended to a wonderful friendship. The ad-hoc scientific chats with Dr. Manh Cuong Pham in the halls of the Chair, slowly grew in a joyful companionship in sports, partying and research. Many thanks are to my “rommie” Dominik Renzel, who I have shared office with and with whom it was always pleasure to collaborate. The last years at the Chair have been filled with even more good moments and laughter thanks to Petru Nicolaescu. He proved to care about my dissertation by being critical while proof-reading this document and listening to my many final talk rehearsals. Many thanks also to Daniele Gloeckner, Claudia Puhl and Gabriele Hoeppermanns for helping me cut through all the red tape that comes with the workplace.

I have been fortunate to always have students around with a genuine interest in mobile cloud computing research. In particular, I appreciate the collaboration with Gökhan Aksakali, Tian Yu, Ke Li, Roman Brandt, and Ghislain Manib Mbogos. Many thanks to Reinhard Linde for all his help in solving hardware and software issues of my projects. I would like to thank Tatjana Liberzon for all she has done to create a relaxed work environment at the Chair. I would like to thank Jia Lia from Australia who volunteered to proof-read the dissertation.

My time in Aachen would not have been so lively and fulfilling without my friends Neda, Bratan, Carlos, Vicki, Stefan, Ina, Ricardo, Yiwei, Pham, Petru, Ana and Zdravko. I also express my gratitude to Emilija Petrovska, who made many compromises to let me do my Ph.D. in Germany.

From the bottom of my heart, I thank my dearest Aleksandra Bogojeska for the encouragement and for the inspiring words to keep going forward she has given to me over the last year. Without her enormous understanding and love this dissertation would not have been completed.

Last but not least, I would like to thank my parents for their love, encouragement, and support throughout my education, which was, especially in the very beginning, not always easy. Thanks to my father for inspiring me to achieve great things in life. Thanks to my mother for being the pillar of support and source of motivation. Words cannot express my gratitude to my brother Spase and sister-in-law Tanja for their unconditional support. You have always been a role model to my life – thank you for making me enormously happy with your “Ph.D.”, i.e. my sweet nephews Ilian and Adrian. Eventually, I hope to have a chance to return to my family what they have given me.

Aachen, May 20, 2014

Dejan Kovachev

*To my mother, for her inspirational courage and persistence, and
to my father, for his kind and benevolent soul.*

Contents

1	Introduction	1
1.1	Problem Description	2
1.2	Research Questions	4
1.3	Research Methods	5
1.4	Thesis Contributions	6
1.5	Thesis Outline	7
2	Research Context	9
2.1	The Need for Cloud Computing	9
2.1.1	Example Application Scenarios	10
2.1.2	Cloud Computing Opportunities	11
2.2	Definition and Taxonomy of Cloud Computing	15
2.2.1	Cloud Access Types	16
2.2.2	Delivery Models	17
2.3	Cloud Computing Technology	19
2.3.1	Basic Underpinnings	19
2.3.2	Multimedia Cloud Computing	26
2.4	Mobile Cloud Computing	28
2.4.1	Cloud Principles Applied Mobile Computing	28
2.4.2	Mobile Cloud Approaches	30
2.5	Multimedia Aspects of Mobile Information Systems	32
2.5.1	Multimedia Metadata	32
2.5.2	Mobile Real-time Communication	36
2.5.3	Mobile User Experience	39
2.5.4	Other Multimedia Aspects	40

CONTENTS

2.6	Community Information Systems	42
2.6.1	Media-centric Theory	45
2.7	Summary	48
3	State of the Art	49
3.1	Cloud-based Multimedia Systems	49
3.1.1	Multimedia Processing in the Cloud	50
3.1.2	Cloud-aware Multimedia	53
3.2	Mobile Cloud Computing	55
3.2.1	Traditional Mobile Computing Models	56
3.2.2	Application Models for Mobile Cloud Computing	57
3.2.3	Comparison of Mobile Cloud Application Models	64
3.3	Mobile Multimedia	67
3.3.1	Metadata Collaboration	67
3.3.2	Communication for Collaborative Applications	68
3.3.3	Mobile Multimedia User Experience	69
3.4	Experiences from Building Mobile Multimedia Community Services . . .	72
3.5	Summary	75
4	Mobile Multimedia Cloud Computing	77
4.1	Faceted View of Mobile Multimedia Clouds	77
4.1.1	System and Technology Facet	78
4.1.2	Mobile Multimedia Facet	81
4.1.3	User and Community Facet	83
4.1.4	Summary of Facets	85
4.2	Application Reference Models	85
4.2.1	“Cloudified” Server	87
4.2.2	Cloud-supported Augmentation	93
4.2.3	Fog/Edge Computing	100
4.3	Summary	101

5	CAELUS: A Cloud Architecture for Enabling Mobile Multimedia Services	105
5.1	Key Requirements	105
5.2	Design Considerations	107
5.2.1	A Commercial Cloud Versus a Custom Test Bed	107
5.2.2	Public Versus Hybrid Cloud Strategy	108
5.2.3	Cloud Interoperability	108
5.3	System Overview	109
5.3.1	The i5Cloud Platform Test bed	109
5.3.2	Mobile Offloading Middleware	113
5.3.3	Core Multimedia Services	115
5.4	Summary	119
6	CAELUS-based Mobile Information Systems	123
6.1	Cloud Platform	123
6.1.1	CIViTra: Cloud Video Transcoder	124
6.1.2	MVCS: Improvement of User Experience for Mobile Video	128
6.1.3	MACS: Adaptive Computation Offloading into the Cloud	139
6.2	Mobile Multimedia	147
6.2.1	AnViAnno: Ubiquitous Multimedia Management	149
6.2.2	XMMC: Collaborative Metadata Services in Cultural Heritage	149
6.3	Personal Cloud Computing	166
6.3.1	Learn-as-you-go: Personal Clouds for Learning	167
6.3.2	DireWolf: Distributed Web Interfaces for Device Clouds	175
6.4	Summary	192
6.4.1	Discussion	193
6.4.2	Achievement of Facets	195
7	Conclusions and Future Work	199
7.1	Summary of Results and Contributions	199
7.2	Future Work	202
	Bibliography	205

CONTENTS

List of Figures	237
List of Tables	241
Appendices	243
Appendix A List of Abbreviations	243
Appendix B Curriculum Vitae	247

Everything is vague to a degree you do not realize till you have tried to make it precise.

Bertrand Russell (1872-1970)

Chapter 1

Introduction

As the world is moving towards connected devices, the request for mobile multimedia services on heterogeneous devices is becoming higher. Several technological trends have contributed to blur the lines between the production and consumption of digital content. Wireless broadband Internet connectivity has become commonplace; and mobile phones have been transformed into digital Swiss-army knives equipped with a multitude of applications that can achieve a variety of different tasks. The armies of users wanting to share memories and the battalions of small communities with highly heterogeneous needs use the mobile Web in many creative ways. Since it is significantly cheaper and more convenient, mobile multimedia has boomed in many domains such as storytelling, live event streaming, practice sharing, video chatting, and watching TV anywhere. Advanced mobile applications that enable new ways of interaction with digital objects have also become increasingly important for on-site professional communities. For instance, commodity smartphones and tablets have proven to provide better flexibility and mobility of information than custom-developed (and expensive) devices in many professional fields, e.g. disaster management or military.

These trends have transformed business operations, service provisioning, and everyday practices. However, the massive amounts of user-generated multimedia content may lack quality and social value. Individuals and communities often do not have access to services and tools for easy value adding, ubiquitous content management, resource-hungry multimedia operations, multimedia sharing and collaboration in real time. Moreover, the growth of such mobile multimedia services is hindered by the lack of simplicity and flexibility in the design, development and deployment processes; despite the fact that mobile multimedia is the driving force behind mobile network and Web growth.

At the same time, large-scale data management and processing, dynamic computing environments and acceleration of service development are drivers for *cloud computing*. Cloud computing has emerged as a paradigm that combines many fields of computing in order to optimize information technology (IT) by setting focus on faster time to market entry, agility and cost reduction. The foundation of cloud computing consists of the delivery of services, software and processing capacity over the Internet, reducing cost, increasing storage,

self-automated systems, and decoupling of service delivery from underlying technology.

Although many cloud-based products and prototypes have shown the potential to significantly change the IT world, cloud computing benefits are far from being achieved for mobile applications. Similar motivations that had driven cloud computing, are also driving the adoption of mobile multimedia cloud computing, but many new research challenges must be overcome. Currently, mobile multimedia cloud computing is at a stage similar to where relational databases were in the early 1980s [Lome11]. Some technology exists, but there are many opportunities for improved technology and for turning the mobile cloud computing into a fruitful paradigm.

This dissertation presents research work aimed at the integration of mobile and cloud environments where multimedia artifacts play central role. An information system architecture based on cloud computing principles which facilitates mobile multimedia services is being proposed. The architecture has been realized through several research prototypes. System engineering aspects have been evaluated and the approach has been validated in several domains such as technology-enhanced learning, digital documentation in cultural heritage and human-computer interaction. The following sections of this chapter describe the research problem, present the research questions and explain the research methodology. In the end, the chapter summarizes the contributions and outlines the rest of the dissertation.

1.1 Problem Description

Cloud computing has great potential to overcome current challenges with asymmetric production and use of multimedia materials, but it has its limitations in mobile settings. Most cloud computing research and development is concentrated on the scalability of enterprise applications over the Web and commonly neglects the needs of (mobile) users [KCK11a].

In current development practices and in the literature we have observed tendencies which may lead to unnecessary frictions in the development of professional mobile web multimedia applications. In both desktop Web browser based and mobile native applications (apps¹), there is an asymmetry in multimedia material production, multimedia processing, and multimedia material consumption on mobile devices. Professional or semi-professional services for multimedia management and services are available for free or at very low rates. However, if we have a closer look at mobile devices themselves, it is not possible for technical amateurs to access multimedia materials without highly specialized mobile apps or to share all their multimedia materials via their mobile Web browsers. This is because of the innovation speed in mobile technologies and the lack of standardization. Consequently, great opportunities for mobile devices like real-time collaboration among mobile users and

¹Throughout the dissertation the term “app” is being used to denote a mobile application (or mobile app) which is a software application designed to run on smartphones, tablet computers and other mobile devices.

mobile semantic processing of multimedia materials can only be realized for high costs in specialized apps.

On the cloud computing level, the development of mobile multimedia information systems is hindered by a lack of an open, customizable lightweight infrastructure which provides a set of services for mobile clients to perform multimedia acquisition, sharing and real-time collaborative semantic annotation. While there are infrastructures that support collaborative features, an infrastructure focusing on multimedia semantics that is “prosumed” by multiple collaborators in real-time is missing. Access to multimedia content and metadata are fundamental in order to support collaborative mobile applications.

On the mobile computing level, even if mobile devices’ hardware and mobile networks continue to evolve and to improve, mobile devices will always be resource-poor, less secure, with unstable connectivity, and with constrained energy. Resource poverty is a major obstacle for many applications, and as a result, computation on mobile devices will always involve a compromise [SBCD09]. For example, on-the-fly editing of video clips on a mobile phone is often not possible because of the amount of time and battery life that is required. When dealing with resource-demanding tasks, the same performance and functionalities which are possible on desktop PCs and notebooks cannot be replicated on mobile devices.

On the multimedia computing level, we can observe three classes of mobile discrepancies between users, content and devices. Firstly, professional video content production shifts to higher resolution formats which are not suitable for common smartphones. For example, TV shows, sport games and movies are usually distributed as high-definition (HD) video (1,920 x 1,080 frame size in pixels), or soon, in the Super Hi-Vision standard (16 times sharper picture than HD TV). The resolution of mobile device increases, but their display capabilities will always be constrained due to the physical size of mobile devices. In effect, it means that more information is being captured than being displayed. Secondly, amateur video content shot with smartphones lacks many characteristics that create the aesthetics of professional videos. For example, mobile video shots are often unsteady, without a smooth pan or zoom to the objects of interest and without clear shot segmentation. On the other hand, cinematographers carefully control the camera movement, intentionally control the lightning and edit the content in post production using their expertise. As a result, the final product is much more appealing. Finally, video navigation in mobile applications follows the desktop metaphor. However, the limited screen size and mobile network bandwidth do not allow for fine navigation within a video using the timeline on the touch screen. Consequently, the user experience with mobile multimedia is diminished.

On the communication level, mobile multimedia applications lack support for real-time collaborative work. Typically, mobile devices usage is limited to creating and sharing content, whereas the collaborative operations are performed asynchronously on desktop computers or laptops. But real-time collaboration is necessary in many cases for both on-site professional and amateur communities. These on-site communities are characterized by a high degree of collaborative work, mobility and integration of data coming from many members. Moreover, the bulk of applications still follow the “single user on a single device”

computing model, despite the fact that people increasingly interact with a collection of heterogeneous computing devices attached to their daily lives. Personal computing is no longer confined to a single device. PCs together with commodity smartphones, tablets, eBook readers, gaming consoles and interactive TVs can be united over the Internet to create collaborative multi-device interactive systems which can benefit from the diverse device capabilities. An individual can interact in many different ways with such symbiotic computing environments, consisting of various personal devices.

1.2 Research Questions

The research challenges in my dissertation come from application of cloud computing principles in mobile and multimedia systems. At the beginning of my dissertation, cloud computing was well established for enterprise applications, but little consideration was paid to mobile and multimedia applications. On the one hand, mobile environments pose additional challenges to the typical fixed wired computer environments. They are characterized by fluctuations in network connectivity, constrained resources on mobile clients, limited form factor of devices, etc. Nevertheless, mobile platforms create unique opportunities for mobile applications and services which can be amplified with context-awareness, ubiquitous access and sharing, real-time on-site communication, etc. Multimedia formats and protocols have traditionally been designed for single-computer storage and processing, but with additional consideration they can be redesigned to harvest the full potential of cloud architectures.

In this dissertation I address the following research questions:

Research Question 1 - How does the cloud computing paradigm affect the design of mobile multimedia services? Specifically, how to enable a single person, i.e. a technical amateur, to design and run large-scale multimedia applications with little effort? What are the design considerations from system, mobile multimedia, and user and community perspectives?

Research Question 2 - What are the innovative application models that facilitate cloud and mobile integration? This question seeks to investigate novel mobile cloud computing beyond the traditional client/server approach. The answer to this question requires theoretical modeling of mobile cloud applications and empirical analysis of performance gains, energy efficiency, development support, and solution applicability.

Research Question 3 - What novel classes of mobile information systems are created with mobile multimedia cloud services? In particular, how do we support ubiquitous collaborations between multiple devices, users and communities using multimedia artifacts? Moreover, how can cloud services be employed to improve the user experience with mobile multimedia?

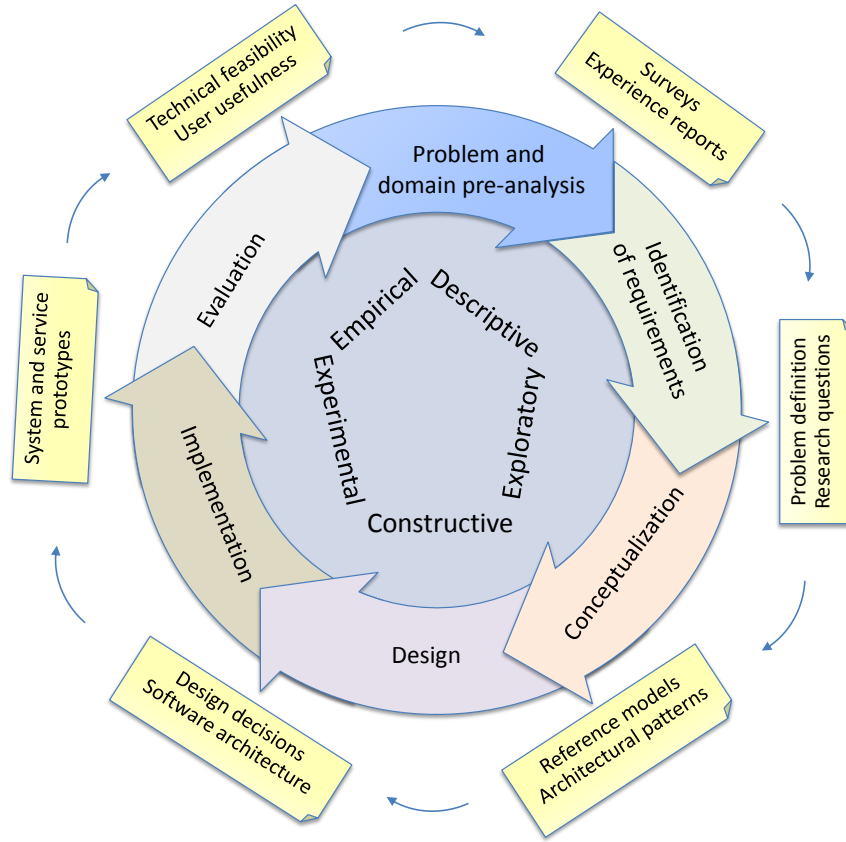


Figure 1.1: Research methods employed in the dissertation

1.3 Research Methods

The objective of the dissertation is not just to solve a single problem, but instead aims to shed light on software architectures, and system and development models that provide the necessary means for solving a variety of tasks involving mobile multimedia operations. A further goal is to examine their applicability bringing forward services for different user groups and communities. To achieve the goals and objectives of the dissertation, several research methods were used. The methods, in general, fall into the categories of descriptive, exploratory, constructive, experimental and empirical research methodologies (see the innermost circle on Figure 1.1). The descriptive methodology helps to describe the research domain state. Experimental and exploratory prototyping of software information systems give insight into novel solutions to relevant practical problems. The constructive methodology favors innovation processes to construct solution ideas, the empirical methods provide ways to collect and analyze data.

The research process featured a spiral cycle of methods comprised of pre-analysis leading to requirements, system conceptualization and design followed by prototypical realization and

evaluation experiments. Precisely, this research work commenced with *survey* studies of related literature on existing approaches. Furthermore, observations and experiences from previous research and development work in mobile and community information systems within our research group framed the identification of three areas of concern (i.e. *facets*) and the definition of key *requirements*. The facets give a comprehensive view on the different perspectives contributing to the use of mobile multimedia cloud computing and the Web. The comprehensive studies resulted in a deepened understanding of the domain in terms of software architectures, processes, and stakeholders involved in the realization mobile multimedia services, which helped to identify and define the problem and research questions. The research process included methods to *conceptualize* mobile multimedia cloud computing and to *design* a comprehensive but multimedia-oriented architecture. Several *prototypical implementations* examined the scope of validity and applicability of the architecture and concepts behind it. *Evaluation* studies on technical and user levels proved the feasibility and usefulness of the proposed approach.

In practice, the research process did not follow in a simple sequence, i.e. it was repeated through several iterations and recursions during the different phases of the CAELUS development. Design, implementation and evaluation phases were often intertwined, i.e. building, testing and observing under different circumstances. Issues with conceptual, abstracted and generalized models were often derived from implementation details.

1.4 Thesis Contributions

Considering the challenge present in mobile multimedia services and the potential of cloud computing paradigm, this dissertation started with the idea to combine both areas. Nevertheless, it is not a simple combination, since many different aspects from various perspectives need to be considered. Besides a complete problem definition and a broad survey on existing related approaches, this dissertation brings forward contributions summarized in the following paragraphs.

First, on the conceptual level I have identified *three facets* that form a general and comprehensive overview of the combination of mobile, multimedia and cloud computing. The areas of concern are grouped into system, mobile multimedia and user and community facets. Each facet covers sub-perspectives that are centered at certain aspects. The system facet consists of data management, communication and computation sub-perspectives which mainly reflect the technological views of mobile cloud systems. The mobile multimedia facet covers adaptation, semantics and modeling for multimedia cloud systems. The user and community facet puts sharing and collaboration, ubiquitous experience and privacy and security aspects of mobile multimedia systems together.

Using these findings, I explained the integration of mobile multimedia services and cloud computing via *three conceptual models*. As shown, in the dissertation, a single model is insufficient to cover all aspects. The reference models cover complementary overviews of

dynamic distributed computing on mobile devices based on assessment of various prototypes. They give general concepts of integration between mobile devices and clouds (computation shifting, cloud computation and storage, programming abstraction, device cooperation, etc.) The first reference model is called “*cloudified*” *server* model alluding to the fact that multimedia services are served from and adapted to the traditional cloud model. The second model accents the mobile device and the ability to opportunistically *augment device’s capabilities* with external cloud resources. This model incorporates context parameters to deliver optimal execution under given constraints and goals. Lastly, the third model pertains to concepts such as *fog computing* and cyber-foraging that envision a shift of small but sufficient cloud resources at the edge of the network close to the end user.

The general facets and the conceptual reference model provided enough insights to derive *key requirements* and an information systems architecture for mobile multimedia services using cloud principles. The architecture is called *CAELUS* which stands for Cloud Architecture for Enabling Mobile Multimedia Services. The realization of CAELUS architecture commences with *i5Cloud*, a test-bed at the levels of an infrastructure and a platform as a service. *i5Cloud* infrastructure provides interfaces to manage virtualized hardware in a convenient cloud-centric way. *i5Cloud* platform components comprise of *core multimedia services* and seamless adaptive computation *offloading middleware*. Taking *i5Cloud* as a foundation and CAELUS concepts as a scaffold, we engineered several research prototypes in the shape of advanced mobile multimedia services and information systems. These prototypes depict the advantages and drawbacks of the approach proposed in this dissertation.

1.5 Thesis Outline

The rest of the dissertation is organized as follows:

- Chapter 2** Chapter 2 introduces background information and defines the used terminology.
- Chapter 3** Chapter 3 gives an overview on the state of the art in the related research areas.
- Chapter 4** In Chapter 4 we propose three facets to analyze the requirements of mobile multimedia cloud platforms systematically. In addition, we provide three reference models for mobile multimedia cloud integration.
- Chapter 5** Chapter 5 depicts the concrete implementation of the model and requirements from Chapter 4.
- Chapter 6** Chapter 6 describes the case studies which demonstrate the validity and applicability of the CAELUS architecture and *i5Cloud* platform and services introduced in Chapter 5.
- Chapter 7** Chapter 7 summarizes the contributions of this dissertation and gives an outlook of still open research opportunities.

As a rule we disbelieve all the facts and theories for which we have no use.

William James (1842 - 1910)

Chapter 2

Research Context

This chapter aims to define of important technologies and terms that coin the research area of mobile cloud computing and re-appear throughout this dissertation. The main ideas derive from cloud computing which as a paradigm shift promotes focused innovation in information and communication technology. This chapter presents the fundamental concepts of this research area. The reasons of cloud's recent emergence are described based on the need for a new computing paradigm, its opportunities and the economics of cloud computing. In addition, the notion of the term cloud computing is highlighted for mobile and multimedia research areas. First, the ideas of cloud computing in mobile settings lead to the definition of mobile cloud computing and types of mobile clouds. Second, the considerations of multimedia aspects of mobile information systems that can be leveraged by cloud computing principles are summarized. Finally, cloud information systems from a community and media theoretic perspective give an additional relevant background information.

2.1 The Need for Cloud Computing

With the rise of modern digital society, the amount of data now available to collect, store, manage, analyze and share is growing continuously. It is becoming common that applications need to scale to datasets of the magnitude of the Web or at least to some fraction of it. Tackling large-data problems is not just reserved for large companies but it is being done by many small communities and individuals. However, it is challenging to handle large amount of data within their own organization budget, time and professional expertise. Software, hardware and communication infrastructure setups cost resources. This is why cloud computing could transform search, mining and analysis tools easily accessible to anyone from anywhere. Large-scale data and cloud computing are closely linked. Moreover, the requirements of businesses, organizations, communities and individuals are rapidly changing, which, in turn, results in a need for constant adaptations of information systems.

2.1.1 Example Application Scenarios

Web Applications

The Web has become a powerful and ubiquitous delivery channel for mass social interaction and collaboration applications. As a result, an enormous quantity of data is produced/consumed every day [BaRa08]. This data comes in forms of content (multimedia), metadata (semantics), structure (links) and usage (logs). Storing, managing, searching and delivering such data volumes raise new challenges; thus motivating development of novel data systems. Tools for indexing, data mining and document/image/video processing are needed. Handling user-generated content imposes problems more related to data volumes, privacy, and delivery latency.

Consider an example of a social networking site. When the number of users on some social networking site starts to grow, then the amount of user-generated data grows too. Increased service popularity means higher request loads. The size of the problem keeps growing, which means increased cost of maintaining and delivering the service. An example for a multimedia processing leveraged in the cloud is a production of multiple versions of the same multimedia artifact. These versions could be different image sizes or image quality (thumbnails or mobile-friendly versions). Multimedia transformation normally does not represent a difficult problem. However, in the case of a popular social network site, size matters (Facebook had 250 billion images as of November 2013). Multimedia processing in reasonable time can only be delivered by using the “unlimited” on-demand computing power of the cloud. Another example is image retrieval from large datasets. Searching the most similar image from a given set of known images requires feature extraction, such as color, texture, histogram, etc. Furthermore, it requires online real time feature comparison between the search item and the dataset. Thus, many innovative Web services require high-end computing resources which were not feasible a decade ago.

The majority of Web startup companies lack the time or resources to focus on their infrastructure. Startup business are accompanied with unpredictable future and limited budgets – they need cost-effective and flexible IT solutions that will enable them to focus on their core processes and business growth.

Business Analytics

Nowadays, business analytics goes beyond finance and accountancy. It helps organizations make better business decisions by figuring out what users like, where they go, who they know, how much time they spend on certain pages. Logging user behavior data through sensors, mobile phones, Web activity generates much data. Large data analysis can help companies to create competitive advantages through learning how to better serve their customers. For example, serving targeted online advertisements requires processing gigantic datasets coming from holiday shopping traffic or media sharing on social network sites.

Normally, processing these dataset on private computing infrastructure would need a couple of days. By leveraging cloud services such as Amazon Web Services (AWS), it is possible to finish the same tasks within couple of hours with a limited budget.

Data Management in Scientific Disciplines

Scientists are overwhelmed with the datasets coming from many different data sources. For example, the Large Hadron Collider at CERN produces about 15 petabytes of data a year for particle physics research. Other examples are large-scale simulations, pharmaceutical drug research, and DNA scanning. In fact, scientific breakthroughs nowadays need technology and tools to manipulate, explore and mine massive datasets. Jim Gray [HTTo09] called it the “fourth paradigm” of science (preceded by the earlier paradigms of theory, experiments and simulations). Scientific computing is revolving around data. Computational science shifts to a data intensive model where scientists analyze observations.

Large data is necessity, not just merely luxury or curiosity. More data leads to better algorithms and systems for solving real-world problems. For example, in the area of natural language processing, Banko and Brill [BaBr01] showed that more data leads to better accuracy for many classification algorithms. Simple features with simple models outperform in most cases sophisticated models based on deep features and less data.

2.1.2 Cloud Computing Opportunities

History has shown that technology shifts have major impact on the business of information technology. The move from mainframe computers to client-server, and then from client-server to the Internet, has been instrumental to innovation in the implementation and utilization of hardware and software. The new levels of abstraction hide the complexity of the code base, resulting in time reduction for developing new ideas. For example, GPU (graphics programming unit) programming in 90’s revolutionized the research and industry when the move from hardware specific implementations by experts to open standards happened. Another similar example is multicore programming with CUDA SDK, OpenCL and DirectCompute which offer a level of abstractions above the graphics pipeline for general purpose GPU computing [SVPS11].

The Evolution of Information Technology Leading to Cloud Computing

The question that arises is “What does the term *cloud* mean?” Over the last two decades people have used cloud figures in information system architectures as a symbol to represent the sum of the Internet, networks, services offered over the Internet, etc. The symbol meant that someone somewhere outside in the world remotely accessed external applications or services. The image of a cloud is a way to depict the enormous potential base of anonymous users.

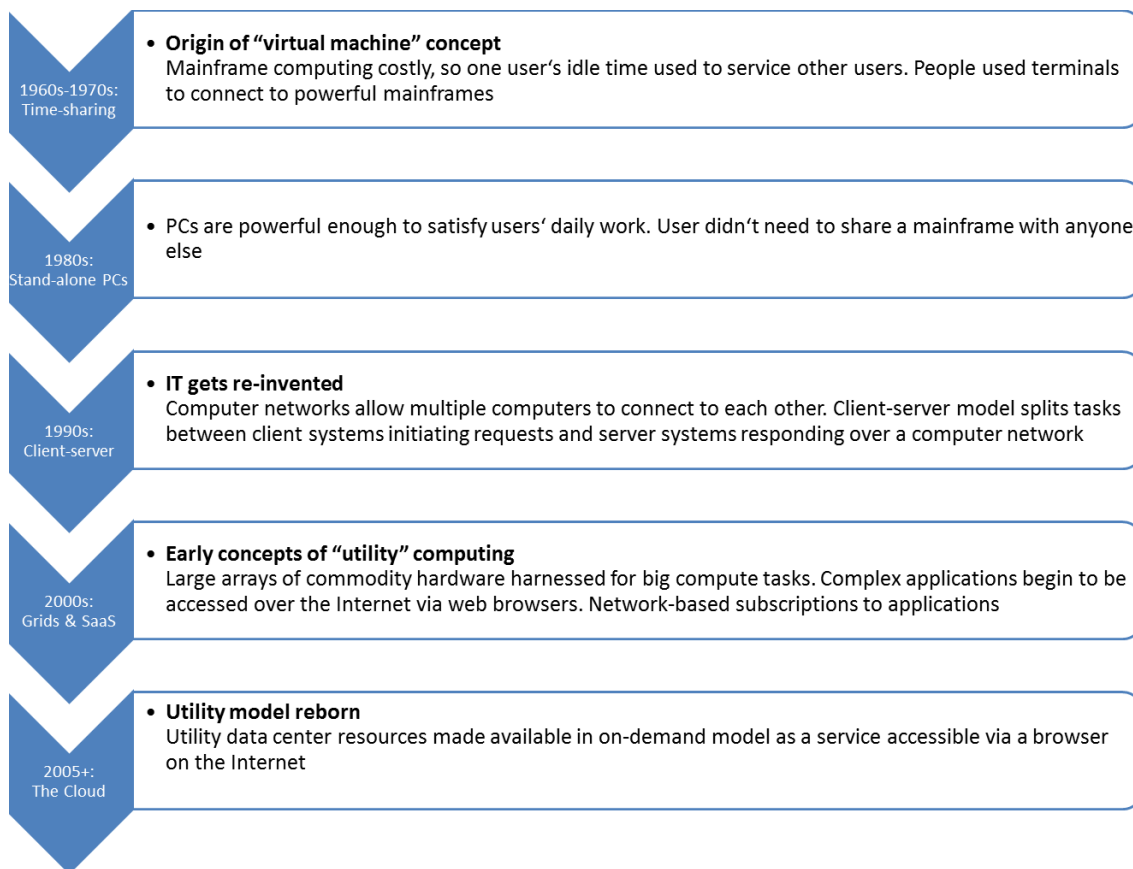


Figure 2.1: Evolution of IT leading to cloud computing (inspired from [RoMa10])

Cloud computing is best understood as an evolutionary change [VoZh09]. As shown in Figure 2.1, its technological and conceptual underpinnings developed gradually over several decades through the various predominant computing paradigms. The technological process spreads over several periods in an evolutionary manner.

In the 1960s, people used terminals to connect to powerful mainframes shared by many users. The terminals consisted of keyboards and monitors. The terminals provided access to the virtualized resources on mainframes. A virtual machine was allocated to individual users, i.e. user seemed to have an entire dedicated machine. This concept of access is similar to the concept of virtualized instances in the cloud, although then a single machine served many users. In the cloud, the actual hardware consists of many thousands of machines. Later in the 1980s, personal computers emerged which were powerful enough to carry on users’ daily tasks. PCs became platforms for thick client applications. Then computer networks emerged which allowed multiple computers to connect and communicate to each other. Resources could be accessed over a local network. The standardization of networking technology simplified the ability to connect local networks to a large global network - the Internet. This has paved the way of software delivery evolution, such as software as a

service and service-oriented architectures. The customer buys access to the software or services for a specified term. The fee scales with the amount of use. Cloud computing providers have later adopted this utility model.

Cloud Computing Economics

Cloud computing is a factor for change and innovation, because of the increased efficiency of information technology infrastructure utilization leads to lower costs. Cloud computing reduces the *cost effectiveness* for the implementation of the hardware, software and license for all.

Large data centers benefit from *economies of scale*. It refers to reductions in unit cost as the size of a facility and the usage levels of other inputs increase [SuSh03]. Large data centers can be run more cost efficiently than private computing infrastructures. They provide resources for a large number of users. They are able to amortize demand fluctuations per user basis, since cloud providers can aggregate the overall demand in a smooth and predictable manner.

The construction and operation of large-scale data-centers at low-cost locations is the key enabler of cloud computing, since they yield factors of cost efficiency by 5 to 7 times in terms of electricity, network bandwidth, software, hardware and maintenance [AFG*09]. These factors, combined with statistical multiplexing to increase utilization form the basics for economy of scale for cloud computing. Cloud vendors can offer cheaper access to infrastructure by purchasing equipment in bulk. They tend to buy much hardware that they can receive discounts much larger than any individual user can achieve. The locations are chosen for their low-cost power and access to fiber optic communications.

Cloud computing helps users achieve *cost reductions*. From users perspective, the opportunities of cloud computing are drawn from the following three mechanisms.

First, the economic appeal of cloud computing is enabling a shift from capital expenses to operating expenses (*CapEx to OpEx*). Because computing resources are paid for as a utility, they can be paid for out of the operating expenditures budget instead of capital investments.

Second, users of cloud computing benefit of *elasticity and transference of risk*. Due to their aggregate and flexible infrastructure, cloud vendors are able to provide customers with resources in a scalable manner. Many user applications can benefit from on-demand resizing of the available infrastructure. Figure 2.2 depicts a typical situation with provisioning hardware infrastructure for applications with variable usage demand. For example, in traditional settings, application owners buy hardware for some predicted demand. Whenever the demand increases, more hardware is needed. There is, however, a lag between the demand increase and time when the new hardware can be made operational. Provisioning computing infrastructure for the peak load leads to underutilization. This could lead to customer dissatisfaction. As a preventive solution, more hardware can be used from the beginning, however, this leads to over-provisioning, i.e. increased operational costs. In

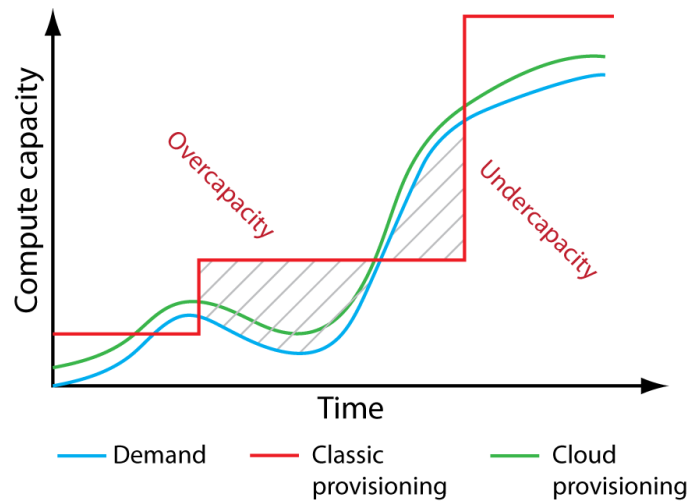


Figure 2.2: Capacity versus utilization with cloud and traditional infrastructure

practice, with this kind of solution one third of servers are only in use for one fifth of the time, to be able to handle peak demands. Another example is when demand is unknown in advance. A new Web service may be hyped at the beginning, followed by reduced demand after some time.

Users can benefit of cloud elasticity by designing the application based on cloud infrastructure. The hardware used can both scale up and scale down rapidly with the usage demand. The owner pays for the actual usage. Thus, the risk of wrong estimating demand is shifted from the service operator to the cloud vendor. The tasks of hardware upgrade, power and networking outages are outsourced to the cloud vendor. The user is alleviated from software patching or version updating. Furthermore, cloud customers can benefit from the “cost associativity” of cloud computing to finish batch operations faster – using 1000 virtual machines for one hour costs the same as using 1 virtual machine for 1000 hours [AFG*09].

Finally, cloud computing empowers organizations with the ability to *focus on core activities*. The absence of up-front capital investment enables users to focus on their core business. Time-to-market is driving force behind cloud computing adoption. Cloud building blocks – such as large-scale geographically distributed storage, databases, elastic IP addresses – allow for rapid prototyping of new functionalities. The ability to rapidly bring new features to market, test their adoption and then improve is a competitive advantage in the ever-changing Internet landscape. Developing for itself capabilities comparable to the cloud building block would require a significant amount of effort. For example, large-scale geographically distributed storage such as Amazon S3 [AmaS3] would have cost a half million dollar investment in labor and servers, assuming to be able to leverage existing open source software.

2.2 Definition and Taxonomy of Cloud Computing

The diversity of technologies related to cloud computing blurs the complete image. NIST [Nist09] defines cloud computing as follows:

“Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

Vaquero et al. [VRCL09] studied many definitions from the literature. They extract a consensus definition containing the minimum characteristics of cloud computing.

“Clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized SLAs.”

The complete definition of cloud computing can only be considered through its main characteristics. Cloud computing differentiates from previous computing paradigms (mainframe, PC, client-server) by its features:

Pay-per-use Model: Cloud computing adopts the utility model. Users obtain computing platforms as easily as other utilities (electricity, water, telephone). They can add or remove capabilities on the fly, without upfront investment in infrastructure and software licenses. The users are entailed to paying only the fee on per usage basis every time the cloud service is used.

Self-healing: In case of failure, there is a hot backup instance of the application ready to take over without disruption (known as failover). It also means that when a developer sets a policy that says everything should always have a backup, when such a fail occurs, the system launches a new backup, maintaining the reliability policies.

SLA-driven: The system is dynamically managed by service-level agreements that define policies such as how quickly responses to requests need to be delivered. If the system is experiencing peaks in load, it will create additional instances of the application on more servers in order to comply with the committed service levels - even at the expense of a low-priority application.

Multi-tenancy: The system is built in a way that allows several customers to share infrastructure, without the customers being aware of it and without compromising the privacy and security of each customer's data.

Service-oriented: The system allows composing applications out of discrete services that are loosely coupled (independent of each other). Changes to or failure of one service will not disrupt other services. It also means one can re-use services.

Virtualized: Applications are decoupled from the underlying hardware. Multiple applications can run on one computer or multiple computers can be used to run one application.

Linearly scalable: It is one of the biggest challenges and ultimate goal. The system is predictable and efficient in growing the application. Ideally, if one server can process 1,000 transactions per second, two servers should be able to process 2,000 transactions per second, and so forth.

Scalable data management: The key to many of these aspects is management of the data: its distribution, partitioning, security and synchronization. New technologies, such as column stores and NoSQL databases augment the large-scale relational database.

Auto-scaling: It enables an increase of resources allocated for some application/client seamlessly during demand spikes in order to preserve systems performance and decrease resources to reduce costs.

In literature, the cloud computing paradigm has been considered from different viewpoints. Depending on the field, the clouds are grouped according to access types, service delivery or business models.

2.2.1 Cloud Access Types

Public Clouds

Public cloud providers offer their resources, such as applications and storage, as services to the general public over the Internet [JeNe10]. The key benefits for the customer are no initial capital investment on infrastructure and shifting of risks to infrastructure providers [ZCBo10]. Customers can set up their application environment fast and inexpensively because the provider takes care of the hardware, application and bandwidth costs. Customers have the ability to access cloud resources how and when they choose. The biggest issue, however, with the public cloud providers is the lack of fine-grained control over data, network and security settings.

Private Clouds

Private clouds are designed for exclusive use by a single organization [Krau09]. They are based on cloud principles where the pooled computing resources are behind a firewall [WSG*09]. They are not available to any users outside the organization. They are built and managed by the organization or by external providers. Pioneering cloud providers like Google and Amazon had initially private clouds for their Web-scale businesses. The advances in virtualization, automation and distributed computing have allowed corporate

data centers to become service providers that can meet the needs of customers outside their corporate boundaries. Private clouds behave similar to public clouds but on a reduced scale. This approach provides highest degree of control over performance, reliability and security. However, it does not provide benefits such as no up-front expenses and newest innovation. Private clouds require a high amount of capital investment. They are affordable only to large organization where the privacy and control is paramount. Using the cloud principles for private cloud applications makes organizations better prepared to migrate or overflow to a public cloud provider when needed.

Hybrid Clouds

Hybrid clouds represent a combination of public and private cloud models that addresses the limitations of each approach. They are used in cases where the capacity of private clouds are exhausted and additional capacity needs to be provisioned outside of the private cloud boundaries. Part of the service infrastructure runs in private clouds while the remaining part runs in public clouds. Amazon Virtual Private Cloud is a secure and seamless bridge between an organization's existing IT infrastructure and Amazon Web Services (AWS) cloud. It enables an enterprise to connect its existing infrastructure to a group of isolated AWS computing resources via virtual private network connection. Furthermore, the organization can take advantage of other AWS management capabilities, such as firewalls, intrusion-detection systems, and security services.

2.2.2 Delivery Models

One common consideration based on delivery model, i.e. the type of IT services that are delivered to the clients. What can be labeled "X as a service" is not clearly determined, where X can take on values such as Hardware, Infrastructure, Framework, Application, Database, and even Datacenter. Cloud technologies are evolving as different vendors try to provide services populating the cloud ecosystem. The most prevalent grouping given by NIST [Nist09] distinguishes three general delivery models: infrastructure as a service, platform as a service and software as a service.

The diagram on Figure 2.3 illustrates the classification of cloud services based on the prepackaged bundles of services which are offered by cloud providers. The leftmost stack represents a private infrastructure owned by the organization/customer. It provides privacy and control over own data and software/hardware, however, it provides largest complexity too. On the rightmost stack, there is less flexibility but also less complexity to be managed.

Infrastructure as a Service (IaaS)

IaaS features on-demand provisioning of infrastructural resources, usually in terms of virtual machines (VMs). It "rents" resources such as servers, network equipment, memory, CPU

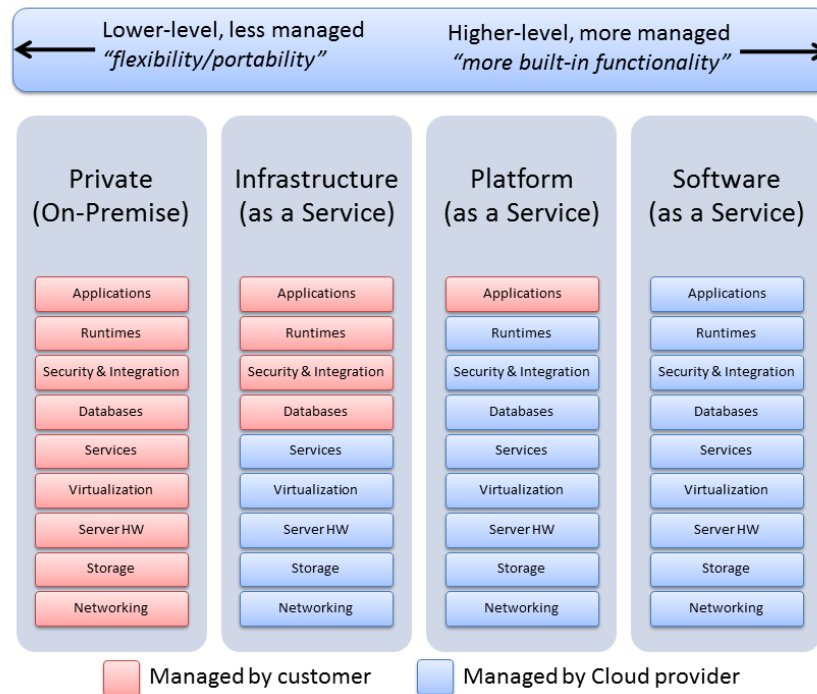


Figure 2.3: "X" as a Service

cycles, storage space, etc. The infrastructure can be scaled up or down dynamically, based on the application resource needs. A good example of IaaS is the Amazon Elastic Compute Cloud (EC2).

IaaS does not provide applications. It simply offers the hardware so that customers can put whatever they want onto it. Rather than purchase of servers, software, racks, and having to pay for the data center space for them, the service provider rents those resources. IaaS proved the lowest level of granularity for managing a cloud applications. Cloud vendors provide VM images prepackaged with different operating systems. The developer has then the opportunity to customize the software stack on the VMs. The management of VMs is exposed through a Web-service Application Programming Interface API or through a Web-based management console. VMs can be brought online and offline when needed. Time it takes to obtain and boot a new server instances is reduced to few minutes, allowing the developer to scale as application needs change. The usage of the VMs is metered by the hour. Storage and bandwidth are also charged by the usage. Typically, storage is metered as gigabyte per month and bandwidth as input/output data traffic.

Platform as a Service (PaaS)

Provisioning and configuring multiple machines for Web applications and data storage can be expensive and time-consuming. PaaS provides platform-layer resources, including

operating system support and software development frameworks. It provides the software platform where applications can run. Developers write application code, and PaaS takes care of the rest. It absorbs the spikes in demand automatically by replication and load balancing. A well known example is Google App Engine [GAE11]. PaaS is similar to IaaS in the way computer “utilities” (CPU consumption, bandwidth, storage) are metered. However, PaaS differentiates from IaaS in a way that applications/developers need less interaction with VMs, e.g. configuration with OS, scaling storage or network throughput. The platform abstracts away this complexity. Developers can easily integrate with other offered services like authentication and email. However, PaaS limits the flexibility. A downfall to PaaS is a lack of interoperability and portability among providers. Developers are constrained by the programming language, the storage engine and database they can use. Moreover, this causes vendor lock-in, i.e. the applications cannot be migrated easily from one provider to another because of API incompatibility.

Software as a Service (SaaS)

SaaS is a model in which an application is hosted as a service to customers who access it via the Internet. SaaS enables provision of on-demand applications and services over the Internet. An office application offered as a Web application such as Salesforce.com [Sale13] is an example. This mode alleviates the customer’s burden of software installation, maintenance, and purchase of software licenses. SaaS was actually the precursor of cloud computing. It was a result of the shift in business model where traditional enterprise license model was turned into pay-as-you-go manner.

2.3 Cloud Computing Technology

2.3.1 Basic Underpinnings

The following section lists basic technologies needed to build a cloud and to understand its working principle. Any application needs a model of storage, a model of computation, and a model of communication [AFG*09]. These models are abstracted in software frameworks for distributed processing of large datasets. They are designed to scale up from single servers to thousands of nodes. In addition, the cloud paradigm is enabled by large data centers that exhibits economies at scale. Building such data centers efficiently arose due to many technological advancements but most notably because of the advancements in virtualization technology.

Data Center

Data center is the physical home of large collection of servers, networking and communications equipment. They are co-located because of their common environmental requirements

and physical security needs, and for ease of maintenance [BaHo09]. Data center size can range as from one room to a whole building. Usual structure is comprised of servers and storage drives mounted in rack cabinets. Furthermore, they can be packed and shipped as containers, which eases the resizing of the data center capacity. Servers in data centers require large bandwidth to and from network backbones. The location is carefully chosen to a geographic proximity to heavy usage areas to keep network latency to a minimum.

World's largest data center owners can build data centers at a fraction of the cost per CPU operation through volume purchasing which can bring huge volume discounts. Servers dominate data-center costs. This is why large cloud vendors rely on custom server construction with tens of thousands of cheap computers with conventional multi-core processors. They are cheaper and more energy-efficient. One powerful machine costs more than two not-so-powerful machines that have the same performance. Having a larger number of small computational units gives an easier way of tackling the fault-tolerant issue. The large data centers are typically located geographically near access to cheap electricity power. The access to natural cooling is essential too. Since the large collection of computer equipment produces a lot of heat, an effective air-conditioning is needed. Data centers must be able to handle power outages (batteries, diesel generators), hardware failure, to avoid loss of data and to minimize the loss of service. Therefore, they must be designed to gracefully tolerate large numbers of component faults with little or no impact on service level performance and availability.

Virtualization

Hardware-assisted virtualization is a key technological enabler for cloud computing. Without virtualization, and the 60+ percent server utilization it allows, the economics of the cloud would be unfeasible. A growing trend in the IT world is virtualizing servers. That is, software can be installed allowing multiple instances of virtual servers to be used. In this way, you can have half a dozen virtual servers running on one physical server. Virtualization allows multiple operating systems to run on a single hardware device at the same time by more efficiently using system resources, like processors and memory. Virtualization enables pooling of computing resources from clusters of servers and dynamically assigning or reassigning virtual resources to applications on-demand.

The virtual machine forms a layer that separates the operating system from underlying platform resources. It hides the physical characteristics of a computing platform from users, and shows another abstract computing platform; thus decouples software from hardware. The OS treats each VM like a physical computer having its own dedicated resources. Virtualization achieves high-server utilization. It divides resources and distributes them to each virtual machine, and it smooths out the variations between applications that need barely any CPU time and those that are compute-intensive. Consequently, it is possible to multiplex many users on a single server and run multiple virtual computers on one physical box.

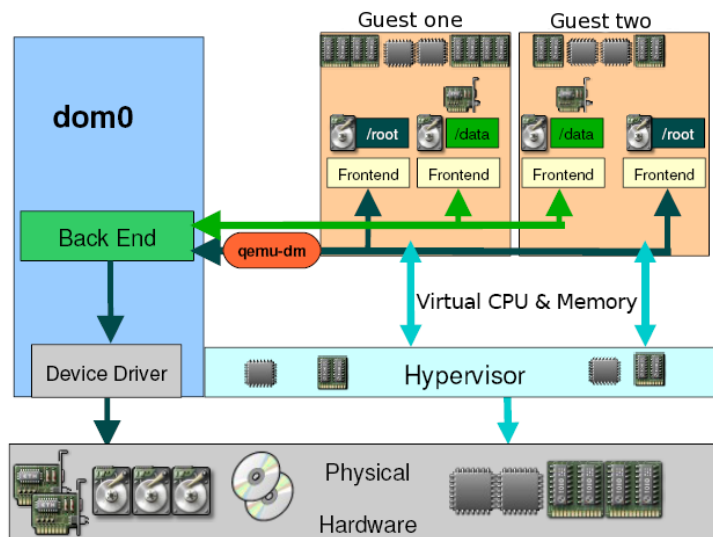


Figure 2.4: Software architecture of a virtualized server (adapted from Xen Hypervisor [XenH13])

Figure 2.4 shows Xen hypervisor runs underneath a Linux operating system which acts as a host managing system resources and virtualization APIs. The host is referred to as dom0 or Domain0. It transforms or “virtualizes” the hardware resources (CPU, RAM, hard disk, and network controller) of a computer to create a fully functional virtual machine that can run its own operating system and applications like a physical computer. This is accomplished by inserting a thin layer of software directly on the computer hardware that contains a virtual machine monitor - a hypervisor. The hypervisor is responsible for allocation of hardware resources dynamically and transparently.

Migration of virtual machines is a key feature of virtualization as software is completely separated from hardware. Migration is useful for:

- Load balancing - guests can be moved to hosts with lower usage when a host becomes overloaded.
- Hardware failover - when hardware devices on the host start to fail, guests can be safely relocated so the host can be powered down and repaired.
- Energy saving - guests can be redistributed to other hosts and host systems powered off to save energy and cut costs in low usage periods.
- Geographic migration - guests can be moved to another location for lower latency or in serious circumstances.

Migration can be performed in two ways with respect to transparency. An offline migration suspends the guest, moves an image of the guests memory. The guest is then resumed on the destination host. A live migration begins moving the memory without stopping the guest.

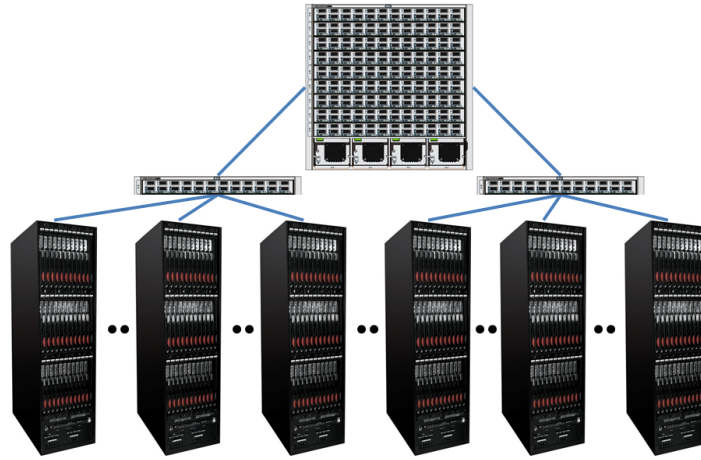


Figure 2.5: Physical organization in a datacenter (adapted from A. Joseph)

All modified memory pages are tracked and sent afterwards. The memory is updated with the changed pages.

Networking Model

The physical organization of data center infrastructure is composed of computing nodes organized into racks (8-64 on a rack). Choosing the network structure involves a trade-off between speed, scale, and cost. The nodes on a single rack are connected by a Gigabit Ethernet network switch which is more of a commodity component. Racks are connected by another level of network or switch with higher number of ports and speed which tend to be much more expensive. In such a network, programmers must be aware of the relatively scarce cluster-level bandwidth resources and try to exploit rack-level networking locality, complicating software development and possibly impacting resource utilization [BaHo09] (see Figure 2.5).

Cloud IaaS vendors typically provide higher-level interfaces to managing the application network through a declarative specification of IP-level topology with internal placement details concealed. Elastic IP addresses allow to allocate a static IP address and programmatically assign it to a VM. Security groups enable restricting which nodes may communicate. Abstraction zones provide abstraction of independent network failure [MVMc10]. In contrast, cloud PaaS typically have fixed topology to scale up and scale down applications automatically and concealed from the programmer.

Storage Model

Innovative Web applications need to handle a large amount of data quickly. To deal with large-scale data, new software stacks are developed. Most basically, the magnitudes of

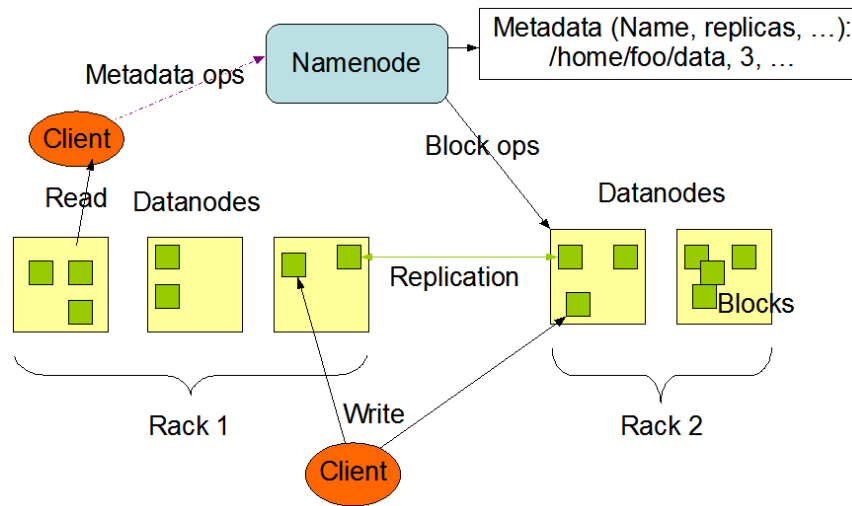


Figure 2.6: Hadoop distributed file system architecture (adapted from [HDFS])

the data require new storage and computation models. At the basic level, large-scale data requires different a file system for storing large files. Files are enormous in size (terabytes). In this case, distributed file systems (DFS) are the typical preferred solution.

The typical distributed file systems in practice are the Google File System (GFS) [GGLe03] and Hadoop Distributed File System (HDFS) [HDFS]. HDFS is an open source of the original GFS. In the implementation, the DFS are usually implemented as follows. Files are divided in chunks of about 64 megabytes in size. Chunks are replicated, typically 3 times at three different computing nodes (data nodes) on different racks, so all copies are not lost after a rack failure. There is a master node (or name node) which is used to index the file chunks. The master node performs operations like opening, closing and renaming files and directories. It determines the mapping of blocks to data nodes. The master node is replicated too. The locations of the master node replicas are contained in a directory which can be replicated too. The users of the DFS know where the directory copies are. The data nodes perform operations like read from and write to file system of data chunks for the clients. Figure 2.6 suggests such architecture.

Distributed files systems have several advantages over single files systems which are enabler features for cloud computing. Hardware failure is anticipated in the software architecture. One HDFS cluster can consist of hundreds or thousands of data nodes, meaning that the probability for failure of some component is relatively high. The DFS are designed to detect failures and to recover automatically. DFS are able to host large datasets is sizes of terabytes. DFS provide interfaces for the cloud applications to move computation code closer to the data since moving computation is cheaper than moving large data volumes. However, DFS is not needed if data and computation fit on one machine. DFSs are designed for batch operation with high data throughput but not for low access latencies.

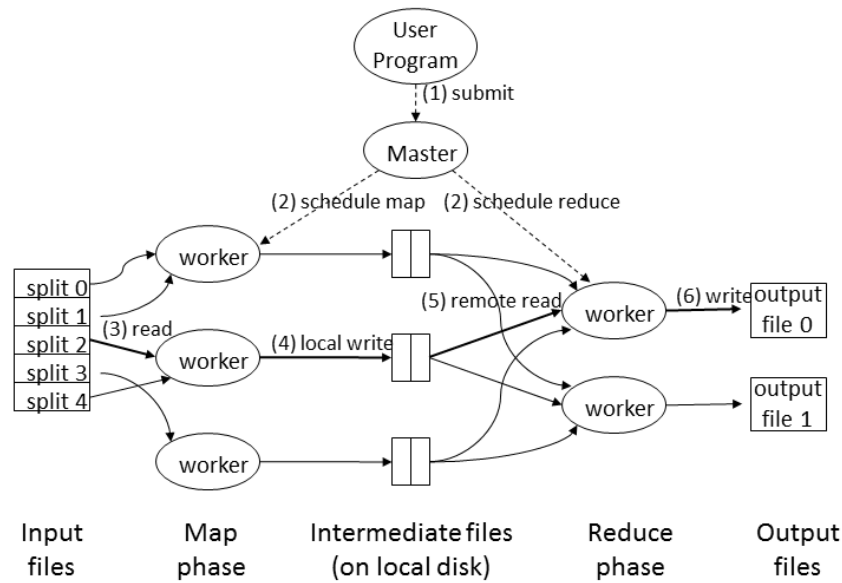


Figure 2.7: The flow of a MapReduce computation (adapted from [DeGh04])

Computation Model

Previously, applications that required large-scale processing, such as scientific computations, were done on special-purpose computers with many processors and specialized hardware. However, the explosion of large-scale Web applications everywhere has caused new, more cost-efficient systems with commodity hardware to be developed. These systems exploit the power of parallelism and at the same time provide reliability at software level. Instead of using expensive hardware, the system takes advantage of thousands of inexpensive independent components with anticipated hardware failure. The software is responsible for ensuring data replications and computation predictability.

On top of the distributed file systems, higher-level programming abstractions are developed. The most popular one is a computational model called *MapReduce* [DeGh04]. It is a software framework introduced by Google in 2004 to support simplified distributed computing on large data sets on clusters of computers. MapReduce programs are designed to compute large volumes of data in a parallel fashion and handle hardware failures during computation.

The main goal of MapReduce systems is to simplify the processing of immense amounts of data quickly. The programmer needs to write two functions, called *Map* and *Reduce*. The MapReduce middleware manages the scheduling, data distribution, communication, parallel execution and synchronization of computational tasks on a cluster of computer machines. It handles errors and failures of task execution too.

MapReduce computation flows as follows (see Figure 2.7). Each Map task is given one or more chunks from a distributed file system. These Map tasks turn the chunk into a sequence of key-value pairs. The way key-value pairs are produced from the input data is determined

by the code written by the user for the Map function. The key-value pairs from each Map task are collected by a master controller and sorted by key. The keys are divided among all the Reduce tasks, so all key-value pairs with the same key wind up at the same Reduce task. The Reduce tasks work on one key at a time, and combine all the values associated with that key in a certain way. The manner of combination of values is determined by the code written by the user for the Reduce function. The steps are listed as follows:

1. The user program forks a Master controller process and some number of Worker processes at different computing nodes.
2. The master creates some number of Map tasks and some number of Reduce tasks, these numbers being selected by the user program. These tasks will be assigned to Worker processes by the Master.
3. Each Map task is assigned to one or more chunks of the input file(s) and executes on it the code written by the user. It is reasonable to create one Map task for every chunk of the input file(s).
4. The Map task creates a file on the local disk of the Worker that executes the Map task.
5. When a Reduce task is assigned by the Master to a Worker process, that task is given all the files that form its input.
6. The Reduce task executes code written by the user and writes its output to a file that is part of the surrounding distributed file system.

MapReduce systems, however, do not replace relational database management systems (DBMS). MapReduce systems are more like an extract-transform-load system than a DBMS, that can quickly load and process a large amount of data in an ad-hoc manner. As such, it complements DBMS technology rather than competes with it [SAD*10]. MapReduce and other related software frameworks emerged as a result of the complexity in programming applications for large-scale clusters.

The MapReduce framework has its own limitations. Its only one input and two stage data flow (map and reduce) are extremely strict. Performing tasks like a different data flow (e.g. joins) are costly. Custom code has to be written for common operations (e.g. filtering) which leads to the fact that the code is usually difficult to reuse and maintain. Moreover, many programmers could not be familiar with the MapReduce framework and would prefer to use SQL. SQL is a high-level declarative language to working with task while leaving all of the execution optimization details to the back end engine. Some example languages such as Pig Latin [GNC*09] emerge between the high-level declarative querying model of SQL and low-level programming using MapReduce. Sawzell [PDGQ05] is a scripting language used at Google on top of MapReduce. A Sawzell program defines operations on a single record of the data. SCOPE [CJL*08] is a high level scripting language for writing large-scale data analysis jobs.

Hybrid systems try to combine the scalability advantages of MapReduce with efficiency and performance advantages of parallel databases. Hive [TSJ*09] is an open-source data warehousing solution built on top of Hadoop. Hive has been built and used by the Facebook data infrastructure team. Hive supports queries expressed in a SQL-like declarative language – HiveQL, which are compiled into MapReduce jobs executed on Hadoop. Hive provides a table-based abstractions over HDFS and make it easy to load structured data. Hive can only run MapReduce jobs and is suited for batch data analysis and also provides a SQL-like query language to execute jobs. The Hive/Hadoop cluster at Facebook stores more than 2 petabytes of uncompressed data and routinely loads 15 terabytes of data daily.

2.3.2 Multimedia Cloud Computing

For long time, high-quality multimedia was reserved to professional organizations equipped with expensive hardware. The distribution of multimedia was limited to hard copies such as VHS, VCD, and DVD. The development of Web 2.0 [Reil05], inexpensive digital cameras and mobile devices has spurred Internet multimedia rapidly. People can generate, edit, process, retrieve, and distribute multimedia content such as images, video, audio, graphics, etc, much easier than before.

Cloud computing has been criticized as a hype [VoZh09], but it drives innovation which can enable the convergence of mobile and desktop multimedia services. Prior to cloud computing, multimedia storage, processing and distribution services were provided by different vendors with their proprietary solutions. The emergence of cloud computing has an immense impact on the whole life cycle of multimedia. Users can store and process multimedia in the cloud in a “pay-as-you-go” fashion. There is no minimum fee and startup cost. Cloud vendors charge for storage and bandwidth. No installation of media software and upgrade are needed. Expensive and complex licenses are aggregated in the cloud offers. The always-on cloud storage provides world wide distribution of popular multimedia content. The cloud can alleviate the computation of user devices and save battery energy of mobile phones. These can greatly facilitate small amateur organizations and individuals. Cloud computing boils down multimedia sharing to simple hyperlink sharing. Sharing through a cloud also improves the quality of service (QoS) because cloud-client connections provide higher bandwidth and lower latency than client-client connections:

Multimedia computing in a cloud imposes great challenges that are unique compared other enterprise applications. Zhu et al. [ZLWL11] identify several challenges for multimedia computing in the cloud as follows.

Multimedia and service heterogeneity: The cloud should support different types of multimedia and services, such as video streaming, voice over IP, photo sharing and editing, media search, media transcoding and adaptation, and content delivery.

QoS heterogeneity: There are different QoS requirements for different multimedia services. The cloud shall support different QoS provisioning.

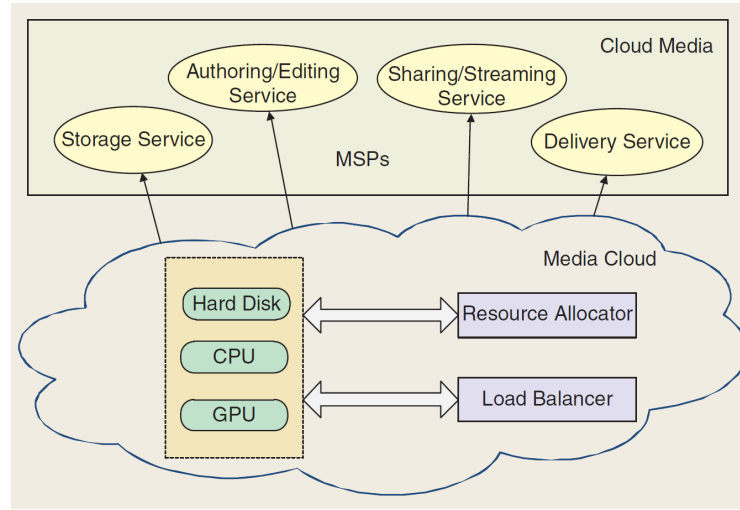


Figure 2.8: The relationship between a media cloud and cloud media (adapted from [ZLWL11])

Network heterogeneity: The cloud needs to adapt the multimedia content for optimal delivery over several networks (wired and wireless) with different network bandwidths and latencies.

Device heterogeneity: Users are equipped with different end devices which have different display, processing, storage and power capabilities. The cloud shall have multimedia adaptation capabilities to fit different types of devices.

In the current state of cloud offerings, the cloud uses utility-like allocation of CPU and storage resources on demand. This is effective for general purpose services like enterprise applications (DBMS or document management systems). In addition of CPU and storage requirements for multimedia services, the QoS in terms of bandwidth, delay and jitter is important, too. Zhu et al. [ZLWL11] consider multimedia cloud computing from two perspectives, i.e. *multimedia-aware cloud* (media cloud) and *cloud-aware multimedia* (cloud media). A multimedia-aware cloud focuses on the provision of QoS for multimedia applications and services. Cloud-aware multimedia focuses on multimedia operations (storage, indexing, search, rendering, etc) in the cloud to best utilize cloud resources.

Figure 2.8 illustrates the relationship between media cloud and cloud media services. The media cloud provides raw resources, such as hard disks, CPU and GPU rented by media services providers (MSP) to serve users. MSPs use media cloud resources to develop their multimedia applications and services.

Alternatively, the roots of multimedia cloud computing can be related with grid computing, content delivery networks (CDN), server-based computing and peer-to-peer (P2P) multimedia delivery. Multimedia computing over grids typically focuses on high-performance computing aspects of multimedia applications [AJNB07]. It applies all computing resources to one single task in order to achieve the maximal performance. Many multimedia applications need large computing power. For example, applications such as image-rendering using

ray-tracing techniques consume a long period of time to give high-quality results, whereas a single powerful computer machine would be insufficient to perform the job. Cloud computing, similarly, applies distributed computing resources to multimedia applications, but unlike grid computing where all resources are applied for a single task, the cloud approach multiplexes many applications on the same infrastructure to achieve maximum utilization and lower costs. The CDN is concerned to minimize the access latency of media files increase bandwidth from the edge of the network. Examples include Amazon CloudFront [AWS] and Akamai [Akam13]. P2P file delivery increases the throughput by offloading the workload to all peers in the network.

2.4 Mobile Cloud Computing

This section tries to answer the question what is mobile cloud computing and how is it related to the generic cloud computing concept. Whereas cloud computing is focused on pooling of resources, mobile technology is focused on pooling and sharing of resources locally enabling alternative use cases for mobile infrastructure, platforms and service delivery. Mobile cloud computing is envisioned to tackle the limitations of mobile devices by integrating cloud computing into the mobile environment.

2.4.1 Cloud Principles Applied Mobile Computing

Resource demanding applications such as 3D video games are being increasingly demanded on mobile phones. Even if mobile devices' hardware and mobile networks continue to evolve and to improve, mobile devices will always be resource-poor, less secure, with unstable connectivity, and with constrained energy. Resource poverty is major obstacle for many applications [SBCD09]. Therefore, computation on mobile devices will always involve a compromise. For example, on-the-fly editing of video clips on a mobile phone is prohibited by the energy and time consumption. Same performance and functionalities on mobile devices still cannot be obtained as on their desktop PCs' or even notebooks' when dealing with tasks containing complicated or resource-demanding tasks.

Mobile devices can be seen as entry points and interface of cloud online services. The combination of cloud computing, wireless communication infrastructure, portable computing devices, location-based services, mobile Web, etc., has laid the foundation for a novel computing model, called *mobile cloud computing* (MCC), which allows users an online access to unlimited computing power and storage space. Taking the cloud computing features in the mobile domain, we define:

“Mobile cloud computing is a model for transparent elastic augmentation of mobile device capabilities via ubiquitous wireless access to cloud storage and computing resources, with context-aware dynamic adjusting of offloading in

respect to change in operating conditions, while preserving available sensing and interactivity capabilities of mobile devices.”

Chang et al. [CGG*13] review other MCC definitions found in the literature and assemble one as follows:

“MCC is an emergent mobile cloud paradigm which leverage mobile computing, networking, and cloud computing to study mobile service models, develop mobile cloud infrastructures, platforms, and service applications for mobile devices. Its objective is to deliver location-aware mobile services with mobility to users based on scalable mobile cloud resources in networks, computers, storages, and mobile devices. Its goal is to deliver them with secure mobile cloud resources, service applications, and data using energy-efficient mobile cloud resources in a pay-as-you-use model.”

Others try to define MCC in many different ways. For example, [Chri09, ZWWe12] define MCC as a combination of mobile Web and cloud computing, i.e. mobile users access the Internet. Mobile cloud computing is, however, more than that combination of cloud computing and mobile environments. Mobile cloud computing brings new types of services which enable not only mobile users to access media and tools, but mobile users to engage in new and rich media experiences that are impossible otherwise. To make this vision a reality beyond simple services, mobile cloud computing has many hurdles to overcome, such as low and variable communication bandwidth, device heterogeneity, security and privacy, etc. The main unique goals behind mobile cloud computing can be summarized as:

- Augment computation and storage capacities. Eliminate existing limitations of current mobile devices.
- Improve energy efficiency and extend battery life. Apply energy-efficient solutions.
- Deliver rich user experience. Support resources-demanding and complex applications and services at low-end devices.
- Achieve development and running of mobile applications and services at low cost. Maximize resource sharing and reuse existing cloud resources.

For example, OnLive [OnLive] is a mobile cloud service that executes video games in a cloud and delivers a video stream to resource-poor clients without interrupting the game experience. In normal cases, playing graphics-intensive games is intended for specialized hardware consoles or high-end PCs, which prohibits playing them on mobile clients. However, the cloud leverages this issue by performing the heavy processing and large memory demands. In the end, mobile players have comparable game experience with a limited battery impact. Many other examples of cloud-based augmentation of mobile devices can be envisioned, e.g. virus scan, mobile file system indexing, and augmented reality applications.

2.4.2 Mobile Cloud Approaches

Mobile cloud computing is also a paradigm shift that drives innovation unique to the classical cloud computing model. Typical cloud computing tools tackle only specific problems such as parallelized processing on massive data volumes [DeGh04], flexible virtual machine management [AWS] or large data storage [CDG*08]. However, the full potential of mobile cloud applications can only be unleashed, if computation and storage are offloaded into the cloud, but without restraining user interactivity, introducing latency or limiting application possibilities. The applications should benefit from the rich built-in sensors which open new doorways to smarter mobile applications. As the mobile environments change, the application has to shift computation between device and cloud without operation interruptions, considering many external and internal parameters. The mobile cloud computing model needs to address the mobile constraints in success to supporting “unlimited” computing capabilities for applications. Such a model should be applicable to different scenarios. The research challenges include how to abstract the complex heterogeneous underlying technology, how to model all the different parameters that influence the performance and interactivity of the application, how to achieve optimal adaptation under different constraints, and how to integrate computation and storage with the cloud while preserving privacy and security.

Achieving MCC benefits can be accomplished in many different ways. Since more layers of communication mechanisms in mobile environments are involved (wired and wireless), cloud infrastructure can be applied at different places in the communication network. Figure 2.9 shows a basic architecture in a MCC scenario.

Client/server software architecture has been widely applied in MCC too. This common software architecture benefits from the cloud by having service backends in a cloud whereas mobile native or Web applications access them over the Internet. In this way, the services use the large computing and storage power of clouds, in addition of having centralized data and user management facilities. Developers could apply same skills and knowledge as of developing typical client server applications. In contrast, cloud infrastructure and services can also be deployed at Internet service providers or even at mobile based stations or at wireless access points. Putting computing power and storage nearer the mobile users results in smaller communication latencies and higher service interactivity, but it has limited scalability. Each architectural choice brings benefits and drawbacks. Chapter 4 elaborates these trade-offs in more details.

Beside the client/server architecture, several other application models gain importance for the MCC research field. These models consider the goals of MCC directly in the software architecture. Here are some examples from the research literature. We consider here only software-based approaches to augment the shortcomings of mobile devices.

Computation offloading or remote execution has gained big attention in mobile cloud computing research, because it has similar aims as the emerging cloud computing paradigm, i.e. to surmount mobile devices’ shortcomings by augmenting their capabilities with external

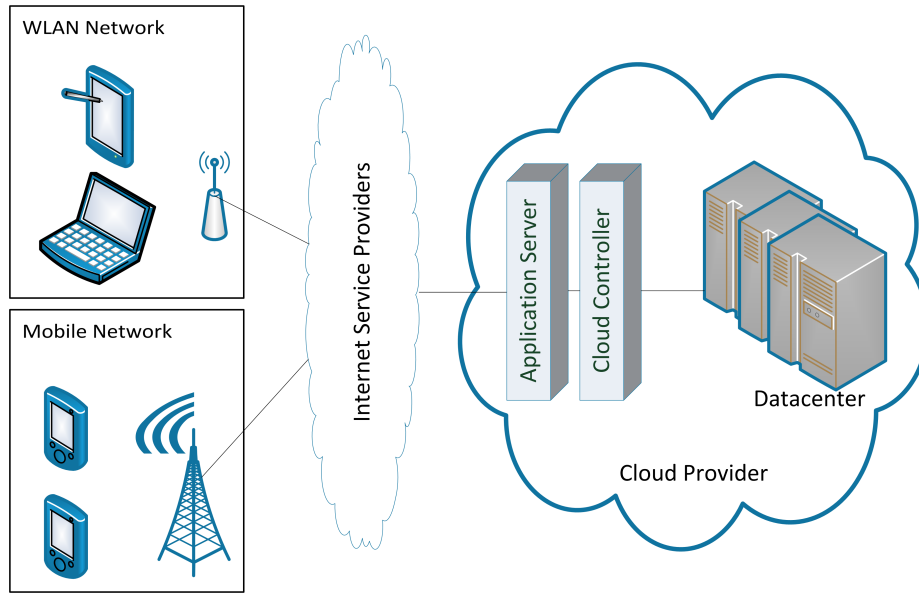


Figure 2.9: Basic mobile cloud architecture

resources. Offloading or augmented execution refers to a technique used to overcome the limitations of mobile phones in terms of computation, memory and battery.

Offloading is a different approach to augmenting mobile devices' capabilities compared to the traditional client/server model prevalent on the Web. Offloading enables mobile applications to use external resources adaptively, i.e. different portions of the application are executed remotely, based on resource availability. For example, in case of unstable wireless Internet connectivity, the mobile applications can still be executed on the device. In contrast, client/server applications have static partitioning of code, data and business logic between the server and client, which is done in development phase. Actually, client/server applications can be seen as a special type of offloaded applications.

Remote data staging refers to the process of extending storage capacity of mobile device using cloud resources. Data staging opportunistically fetches and caches data from remote storage [FSTS03]. The main challenge is to reduce file access latency for interactive applications instead of suffering from full Internet latency or limited device storage. Mobile applications need to have synchronized data between multiple wired and wireless devices. Data staging techniques try so abstract the complexities of networking data management issues for developing mobile applications [CCS*12]. The underlying middleware support multiple users and devices, and very importantly, disconnected operations.

Elastic mobile applications can adaptively be split and parts offloaded [KPKB10, ZKJG11]. Basically, this application model provides developers the illusion as if they are programming virtually much more powerful mobile devices than those with actual capacities. Elastic application use offloading techniques to make decisions on how and when to partition a mobile application. Moreover, elastic applications can be partitioned at different granularity,

e.g. at process level or module level. Parts of the application execute in a distributed manner on different hosts. Parts of the application can be shifted dynamically between the source host device, a cloud provider or other devices. This migration decision depends on many parameters, but the main goal is to keep application integrity, and result correctness and consistency. This approach can also be applied to adapt the application fidelity, i.e. adjust quality of application execution depending on availability of local and remote resources.

Ad-hoc mobile clouds refers to the mobile applications and services that can form arbitrary, opportunistic groups of mobile devices, i.e. “clouds” of local resources utilized to achieve a common goal in a distributed manner [FLRa11]. The devices expose their hardware to support a joint shared workload at runtime. This idea has been extensively researched in P2P wired networks of desktop computers and mobile sensor networks, but it has recently attracted fresh attention with the advancement of mobile hardware and communication technology. Compared to offloading, mobile ad-hoc clouds are independent of a remote cloud. They are especially useful in situations with low Internet connectivity and high co-location of mobile devices.

2.5 Multimedia Aspects of Mobile Information Systems

Building mobile information systems relies on many factors to achieve the cloud benefits. Media plays an essential role in the building mobile information systems. Multimedia refers to typically to media components other than text, i.e. the combinations of two or more media types including images, video, animations, 3D models, etc. Multimedia delivery over wireless communication has been largely researched, however, in this section, other important aspects are focused on. This includes user experience, metadata and content streaming, mechanisms for mobile collaboration, and others.

2.5.1 Multimedia Metadata

Multimedia metadata, or descriptive data about the multimedia content contains different facets of multimedia content, e.g. formal and technical properties (e.g. encoding, format, creation details), digital rights, structure and semantics of the contents itself. Metadata allows to tie different multimedia processes in a life cycle together. Kosch et al. [KBD*05] identify two main parts of the metadata space. First, *metadata production* occurs at or after content production. At this stage, the metadata consists of creation information, automatically-extracted information (low-level features such as histograms and segment recognition) and human-generated information (high-level semantics such as scene descriptions and emotional impressions). Second, *metadata consumption* occurs at the media dissemination and presentation stages of the content’s life cycle. For example, metadata facilitates retrieval capabilities for large multimedia databases and guides the adaptation of

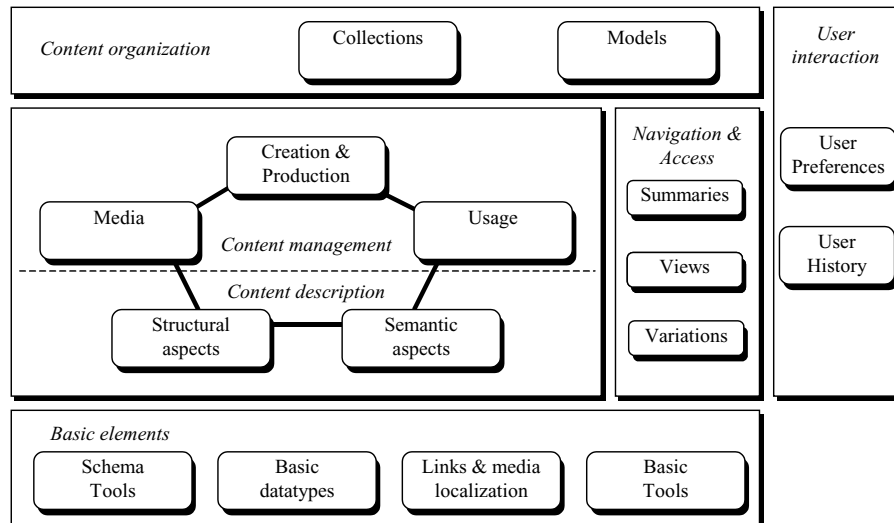


Figure 2.10: Overview of MPEG-7 Multimedia Description Schemes (adapted from [ISO03])

content to achieve the desired QoS. Metadata is consumed at different stages: at authoring, indexing, proxy level, and end-device level.

There are many description standards for multimedia materials such as P_Meta Semantic Metadata Schema [PMet07], SMPT-DMS-1 [SMPT07], Dublin Core [DCMI05], MPEG-7 [Kosc02], ARML [Lech10] and TV-Anytime [TVA05]. In this research work, the MPEG-7 and ARML standards were used.

MPEG-7 provides a rich set of audiovisual description elements to describe multimedia contents and to guide the search and retrieval of digital multimedia content. MPEG-7 can describe audiovisual information regardless of storage, coding, display, transmission, medium, or technology. It addresses a wide variety of media types including still pictures, graphics, 3D models, audio, speech, video, and combinations of these (e.g., multimedia presentations). It doesn't define the encoding of video and audio, but uses XML (Extensible Markup Language) to store multimedia metadata. Based on XML Schema it lets itself be extended easily by using the Descriptor Definition Language (DDL), thus becomes a basis for larger more complex multimedia metadata vocabulary. MPEG-7 has two pre-defined types of elements: Description Schemes and Descriptors. A specific description scheme can be defined by freely combinable descriptors. Each descriptor refers to a specific feature or attribute of multimedia content [Kosc03, ISO04d]. DDL allows the creation of new descriptors and description schemes within the standard.

The MPEG-7 Multimedia Description Schemes (MDS) [ISO03] contain several high-level description schemes suited to describe and manage versatile multimedia and audio-visual contents (see Fig. 2.10). Of a particular interest for mobile multimedia are Visual Description Schemes and Audio Description Schemes which provide descriptors for visual and audio media in relation with spatio-temporal information [Klam11]. Figure 2.10 depicts

other MDS. The Structural Aspects can contain data about physical, spatial and temporal segments. The Conceptual Aspects describe the semantic of the content, such as events, objects, concepts, and their relations. The Usage describe the usage information related to the audio-visual content, such as usage rights, usage record, and financial information. Media holds information on format, coding, instances, size, and resolution etc. Creation Information describes the creation and classification of the AV content and of other related materials. The creation information provides a title (which may itself be textual or another piece of AV content), textual annotation, and information such as creators, creation locations, and dates. Navigation and Access support discovery, browsing, navigation, visualization and variation adaptive to various users. User Interaction describes the user's activity on media. Collection is used for the organization of multiple media. Models provide an analytic base for data processing.

Augmented Reality Markup Language (ARML) is a metadata specification that enables content developers to create content that can be displayed on various mobile AR browsers. ARML is originated from KML (Keyhole Markup Language) by reducing some tags that are not relevant for AR and adding some new features. An ARML example document which is shown at listing 2.5.1 demonstrates the structure of the specification. An ARML file consists of two sections. The first section provides information about the content provider that the POIs are related to. At this section, provider's name, description, logo, URL, icon and related tags are specified. The second section consists of the POIs. A place-mark element corresponds to a POI. As seen every place-mark is linked to the content provider which is defined at content provider section [Lech10]. A place-mark tag, consists of with AR tags such as: name, description, URL, coordinates, e-mail etc. The coordinates tag provides longitude, latitude and altitude information consecutively.

In recent years, the Web 2.0 paradigm fostered the rise of user-created metadata, namely in the forms of collaborative tagging and annotating. They have increased in popularity with arrival of social bookmarking services such as Delicious.com [Deli13b] and photo sharing services such as Flickr.com [Flic13]. When collaborative tagging is applied, a system of organization emerges; an organization called *folksonomy*, where people use their own vocabulary to categorize Web documents. This user-created metadata, where users of the documents and media create metadata for their own individual use, is also shared throughout a community [Math04]. Folksonomies are successful because people are able to directly reflect their vocabulary and because it is much cheaper than building a complex classification hierarchy. However, systems based on folksonomies lack community awareness and community support. It is not possible to specify, in which community context a tag assignment has been defined. Different communities should be able to create different — even contradictory — community-specific terminologies for multimedia contents. Users should be able to form groups and restrict access to group-specific media. These lead all to the conceptualization of *commsonomies*, i.e. cross-media and cross-community sharing of community-specific folksonomies [KSRe07].

Collaboration plays a significant role among a group of people (e.g. coworkers) trying

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
  xmlns:ar="http://www.openarm1.org/arm1/1.0"
  xmlns:wiktitude="http://www.openarm1.org/wiktitude/1.0">
<Document>
  <ar:provider id="mountain-tours-I-love.com">
    <ar:name>Mountain Tours I Love</ar:name>
    <ar:description>My preferred mountain tours in the alps. Summer and Winter.</ar:
      description>
    <wiktitude:providerUrl>http://www.providerhomepage.com </wiktitude:providerUrl>
    <wiktitude:tags>travel, hiking, skiing, mountains</wiktitude:tags>
    <wiktitude:logo>http://www.mountain-tours-I-love.com/wiktitude-logo.png </wiktitude:logo>
    <wiktitude:icon>http://www.mountain-tours-I-love.com/wiktitude-icon.png </wiktitude:icon>
  </ar:provider>
  <Placemark id="123">
    <ar:provider>mountain-tours-I-love.com</ar:provider>
    <name>Gaisberg</name>
    <description>Gaisberg is a mountain to the east of Salzburg, Austria</description>
    <wiktitude:info>
      <wiktitude:thumbnail>http://www.mountain-tours-I-love.com/gaisberg-thumb.png </wiktitude:
        thumbnail>
      <wiktitude:phone>555-9943</wiktitude:phone>
      <wiktitude:url>http://en.wikipedia.org/wiki/Gaisberg </wiktitude:url>
      <wiktitude:email>info@mountain-tours-I-love.com</wiktitude:email>
      <wiktitude:address>Jakob-Haringer-Str. 5a, 5020 Salzburg, Austria</wiktitude:address>
      <wiktitude:attachment>http://www.mountain-tours-I-love.com/gaisberg.pdf </wiktitude:
        attachment>
    </wiktitude:info>
    <Point>
      <coordinates>13.11,47.81,1158</coordinates>
    </Point>
  </Placemark>
  <Placemark id="534">
    ...
  </Placemark>
</Document>
</kml>
```

Listing 2.1: ARML Example

to perform tasks to achieve a common goal, or having similar interests. The bulk of mobile collaborative activities occur using multimedia as the main communication artifact. Social software is just another example of multimedia collaboration — sharing, tagging and commenting on images or videos are the common actions performed. Description technologies such as XML and RDF ease the creation, utilization and maintenance of collaborative metadata. Users are able to collaborate mostly on the multimedia metadata once the multimedia content has been created and shared.

2.5.2 Mobile Real-time Communication

Real-time communication protocols are powerful tool for online collaboration in which all users can exchange data and messages instantly or with negligible latency. Although mobile devices provide interfaces for real-time voice communication, mobile application development tools lack general real-time communication mechanisms to exchange arbitrary text and multimedia data. Instant messaging protocols had early focus on text communication, but with the ubiquity of multimedia-enabled devices, there has been great interest in extending instant messaging to support multimedia interactions [Sain07]. Moreover, video streaming protocols have unique features since the real-time delivery of video content requires large bandwidth and network-based video content adaptation. This section introduces text (i.e. XML-based) and video streaming protocols which were surveyed in a seminar paper [Sadi12]. These protocols have been used extensively in many projects in our group and in this dissertation.

Extensible Messaging and Presence Protocol

The Extensible Messaging and Presence Protocol (XMPP) [SSTr09] is an essential enabling technology for cloud services. XMPP as an instant messaging protocol is able to enhance near real-time multimedia sharing and collaboration on multimedia metadata. The protocol originates from the Jabber open source community that developed it mainly for instant messaging purposes in 2000. Later in 2004 it was turned into a RFC standard by Internet Engineering Task Force. The XMPP Standards Foundation has been a set of extensions to the core protocols which are called XMPP Extension Protocol (XEP). Although Jabber developers just focused on building instant massaging, with the extensible nature of XML enables an extension point for exchanging structured data on a reliable and standard infrastructure. Consequently, XMPP has been used in a wide range of purposes i.e. presence, instant massaging, voice and video calls, real time collaboration and as light-weight middle-ware. In addition to the current extensions, the protocol can be easily extended for further purposes and and these extensions can even be standardized within the community. Beyond that, there exist many open source client and server implementations since the platform is an open standard.

XMPP-based communication takes place not only among various users and their communities, but also among multimedia cloud services. Cloud services can exchange XML-based control messages and data in near real-time. The XMPP protocol provides a pure XML foundation for real-time messaging, opening up tremendous possibilities for more advanced real-time applications. XMPP together with its extensions is a powerful protocol for cloud services. Together they demonstrate several advantages beyond traditional HTTP-based Web services, e.g. Simple Object Access Protocol (SOAP) and Representational State Transfer (REST):

- decentralized, open and flexible (extensible) communication protocol
- services discoverable without the need of an external registry
- federation of services enabling easy weaving of cloud services together
- built-in presence functionality providing resource and availability discovery at runtime
- support for real-time data streaming in two directions
- interoperability with other protocols and programming language independent
- event notifications
- remote procedure calls (e.g. SOAP over XMPP)
- multimedia session management

Many advantages over existing technologies make XMPP a highly interesting candidate for next-generation online services. HTTP was originally designed to accommodate query and retrieval of Web pages and did not aim at rather complex communication. The intrinsic synchronous HTTP protocol is unsuitable for time-consuming operations, like computation-demanding database lookups or video processing. Asynchronous invocation of XMPP eliminates the need for ad-hoc solutions like polling.

An XMPP network can be seen as a complete XML-based routing framework upon which a messaging middleware can be built. Hence, an XMPP-based middleware can be used to integrate different services into a distributed computing environment. The application modules, external sensors and external services are XMPP entities identified by unique JIDs. The exchanged messages through XML are called stanzas. XMPP not only allows discovering of services out of the box, but also supports determining their status and availability. Hornsby and Walsh [HoWa10] give an overview of the development and use of XMPP in many domains related to cloud computing.

Video Streaming Protocols

Real Time Streaming Protocol (RTSP) was developed by the Internet Engineering Task Force [Vetr11]. The main purpose of RTSP is to establish and control media sessions. The transmission of data is not handled by RTSP but is done by Real-Time Transport Protocol (RTP). RTP carries the media stream and data over the Internet. It is a stateful protocol used for streaming audio and video contents. This means that a session has to be maintained between a server and a client while they are communicating. Similarly, the synchronization of client and server needs timely delivery of data contents from each other.

There are some issues related to RTSP. The Internet congestion does not allow RTP to make the timely delivery of the contents. Similarly, it is resource intensive when large number of clients are streaming videos simultaneously from a single server as session is required for each of the clients. Though it works well with managed Internet Protocols, it is not supported by most of the Content Delivery Networks. RTP is also not allowed through firewalls. However, it is possible to pass RTP through firewalls by encapsulating it inside TCP port 80 packets. This causes the network performance to decrease significantly, and increases load on the server.

HTTP Progressive Delivery is the process of delivering video content in a browser via HTTP on port 80. The problem of RTSP being not able to pass through firewalls is remedied with the “firewall-friendly” HTTP [Vetr11]. Furthermore, HTTP is stateless, i.e. no session is required, which eliminates the disadvantage of RTSP regarding the extensive resource usage. The main disadvantage of the progressive delivery is that a user cannot jump to a certain point in a video if that part has not been downloaded yet. The contents are downloaded as chunks, only the header of the file has the metadata required to play the video. So, for a user to be able to view all the contents, the video has to download progressively. In other words, the video is buffered first and played. The buffering is noticeable especially when there is a slow Internet connection. The user has to wait until all that portion of the video is downloaded and the user is able to see the grey bar on the video player. Sometimes, the download is slow than the playing of the video and the video is halted. When there is enough data to be played in the buffer, then only the video playback happens. The progressively downloaded videos can be viewed differently with different media players.

HTTP Streaming is similar to HTTP Progressive Delivery, but with additional functionalities. This includes the ability of client to request certain portion of a video from the streaming server. This overcomes the disadvantage of HTTP Progressive Delivery of not being able to jump forward and backward on a video. With this functionality, HTTP Streaming is suitable for longer videos.

For example, a request for certain fragment of media can be done as:

```
http://www.example.com/example/ogv?t=10,20  
http://www.example.com/example.ogv#track=audio&t=10,20
```

Listing 2.2: Example of media fragment URI

This example shows getting the contents from a Web site for the time range of 10-20 seconds. The second example requests only the audio track for the time between 10 and 20 seconds. As it is similar to the HTTP Progressive Delivery the problem faced by the former of not being able to adapt to degrading network bandwidth is also persistent in this protocol.

Adaptive Streaming adjusts the video and audio quality with slow network connection. It uses HTTP as its transport protocol. In this streaming method, the server stores copies with different qualities of a video content. The videos are then split into the segment size of 2-10 seconds. The server delivers the first chunk using progressive download and the network connection is checked for downloading another chunk. The video quality decreases when the network connection is not very good. In this way, the quality of video and audio starts degrading gracefully and correspondingly with network bandwidth.

Dynamic Adaptive Streaming over HTTP (DASH) is an emerging streaming standard, where the client has control over the delivery. The client is responsible for choosing the alternatives according to the network bandwidth. The multimedia streaming technology that is currently in interest is MPEG-DASH. It is not a system, protocol, presentation, codec, interactivity or client specification but provides a format to enable efficient and high quality delivery of streaming services over the Internet [Stoc11]. In dynamic adaptive streaming, the client has control over the delivery.

2.5.3 Mobile User Experience

For multimedia services, in addition to CPU and storage requirements, another very important factor is the quality of services or quality of experience (QoE). For a long time there was no common agreement on the term user experience (UX). For some people it is only seen as being solely user and interface dependent. This understanding of UX changed during the last few years. The definition of a general user experience definition is very hard. First, it is connected with many concepts like emotional, affective, experiential, hedonistic, and aesthetic variables [HaTr06]. Researchers include and exclude these variables arbitrary, depending on the author's background and interests. Second, the unit of analysis can range from a single user interaction aspect to all aspects including interaction with the company. Third, UX research is fragmented and complicated by various theoretical models.

In 2009 Law et al. [LRH*09] presented a survey on UX to define this term. They designed a questionnaire with three sections: "UX Statements", "UX Definitions" and "Your Background". In "UX Statements" they gave 23 statements and the participants were asked to indicate their levels of agreement using a 5-point scale. Two hundred and seventy five researchers and practitioners from academia and industry were asked to fill out the questionnaire. Despite their predictions there is a relatively common view on the term user experience. Most researchers agree that UX is influenced by the current internal state of the person, earlier experiences and the current context. Additionally, most researchers do agree on the fact that user experience should be assessed while interacting with an artifact. Nevertheless, it has also effects long after the interaction. Finally, in 2010 an ISO

(International Standards Organization) definition of UX has been published and defines UX as “a person’s perceptions and responses that result from the use or anticipated use of a product, system or service” [ISOF10].

Mobile user experience (MEX) is an ongoing field of study. There are few research results most researchers do agree on. Subramanya and Yi [SuYi07], for example, recognize in their study of MEX three dimensions: device-related issues, communication-related issues and application-related issues. Furthermore, Nielsen [Niel09] identifies the mobile user as more distracted, in context and impatient as the browse behavior witnessed on a desktop PC is substituted with search behavior. Offloading CPU intensive tasks into the cloud could overcome problems of slow response and reaction times.

User experience in the context of mobile media is dependent on several aspects. The most obvious one is the small screen size of mobile phones. Even 4 inch displays are small in comparison to today’s LCD TVs (often starting at 32inch). This leads to problems in browsing/seeking videos and watching big screen production as it becomes hard to follow the event in some cases. Therefore, solutions like zooming, ROI, etc. have to be used to overcome these limitations [KPSV07].

Device-related issues affecting the user experience are the limited battery power, changing bandwidth, WiFi handover and others. Furthermore, the attention on mobile video is lower than on the TV. For watching a movie people sit down in front of a TV and can spend 2 hours there without a problem. In comparison to that mobile users are most often in movement. Taking the train to work or other transportation services are often the common situation of mobile phone usage like video consumption. This time is limited and the attention is influenced by the sounds around a person [CCJu07].

2.5.4 Other Multimedia Aspects

Throughout this research work, several multimedia technologies and operations are being used. They are briefly described here.

Multimedia transcoding becomes more common procedure as the interoperability between different media devices becomes more important. One of the biggest challenges in future multimedia application development is device heterogeneity. Future users are likely to own many types of devices. Users when switching from one device to another would expect to have ubiquitous access to their multimedia content. Cloud computing is one of the promising solutions to offloading the tedious multimedia processing on mobile devices and to making the storage and access transparent. Transcoding, generally, is the process of conversion of one coded format to another. Video transcoding, for example, can consist of adapting the bit rate to meet an available channel capacity, or reduction of the spatial or temporal resolution to match the constraints of mobile device screens. Video transcoding and processing are data intensive, time and resource consuming. Clouds play a major role in reducing the costs for upfront investment in infrastructure and in cases of variable demands. Multimedia indexing

refers to the process of multimedia processing in order to identify content objects and cues which can be later used for content-based multimedia retrieval. Indexing solutions usually involve resource-expensive computer vision and machine learning algorithms, which too can benefit from a cloud infrastructure.

Tagging is a powerful and flexible approach to organizing the content and the learning processes in a personalized manner. With the rise of Web 2.0, the word tag has been used in almost every Web 2.0 or Web page. Rather than using a standard set of categories defined by experts, everybody creates one's own categories in the form of tags. Tagging helps users collect, find, and organize multimedia effectively. Tags can be available to all online users, user community groups, or only accessed by the creator privately. Tags are applied to different resources such as images, videos, Web pages, blog entries, and news entries. Various Web resources are organized through tags.

Digital storytelling and story creation. Storytelling intertwines semantic knowledge by linking it with the narrative experiences. Storytelling is an important aspect for knowledge sharing and learning in professional communities. Telling, sharing and experiencing stories are common ways to overcome problems by learning from the experiences of other members. One of the major reasons for the limited adoption of digital storytelling in organizational information systems may be that story authoring is extremely challenging. Suitable tools and simple methodologies need to be put in place to support authors in the use of different media. The development of a shared practice integrates the negotiation of meaning between the members as well as the mutual engagement in joint enterprises and a common repertoire of activities, symbols, and multimedia artifacts. Storytelling and story creation through interactive and effective stories enable joining conceptual and episodic knowledge creation processes with semantically enriched multimedia.

Semantic multimedia retargeting seeks to remedy some of the issues with UX in mobile video applications by making use of cloud services for fast and intelligent video processing. Cloud computing has great potential to leverage the current issues with mobile production and use of multimedia materials, in general, and with mobile UX, in particular. Multimedia processing techniques like automatic video zooming, segmentation, and event/object detection are often proposed techniques for video retargeting to mobile devices. For example, zooming and panning to the regions of interest within the spatial display dimension can be utilized. This kind of zooming displays the cropped region of interest (ROI) at a higher resolution, i.e. observe more details. Panning enables watching the same level of zoom (size of ROI), but with other ROI coordinates. For example, in soccer game this would mean watching how player dribble with the ball more closely, whereas by panning one can observe other players during the game such as the goalkeeper.

Mobile augmented reality is a natural complement to mobile computing, since the physical world becomes a part of the user interface (e.g. in video streaming). Accessing and understanding information related to the real world becomes easier. This has led to a widespread commercial adoption of MAR in many domains like education and instruction, cultural heritage, assisted directions, marketing, shopping, gaming. For example, Google's

Sky Map gives a new and intelligent window on the night sky or CarFinder creates a visible marker showing parked car, its distance away and the direction in which to head. Furthermore, in order to support diverse digital content several popular MAR applications have shifted from special-purpose applications into MAR browsers which can display third party content. Such content providers use predefined APIs which can be used to feed content to the MAR browser based on context parameters.

Recommender systems have emerged as a kind of information filtering systems that help users dealing with information overload. They are applied successfully in many domains, especially in e-business applications like Amazon.com. The basic idea of recommender systems is to suggest to users items, e.g. movies, books, music, etc., that they may be interested in. Since computing is moving toward pervasive and ubiquitous applications, it becomes increasingly important to incorporate contextual aspects into the interaction in order to deliver the right information to the right users, in the right place, and at the right time. Considering the high-level of computational efforts needed to generate recommendations, cloud-based recommenders systems are many time the only viable solution.

2.6 Community Information Systems

Mobile technologies enable new ways of working. Mobility supports collaboration between the members of a organization, helping to create professional and social communities. Wenger et al. [Weng98] defines: “Communities of Practice (CoP) as groups of people who share a concern, a set of problems, or a passion about a topic, and who deepen their knowledge and expertise in this area by interacting on an on-going basis.” The most important processes in a CoP are collective learning and the production of shared meaning and collective identity. The social practice consists of explicit and tacit knowledge, and competencies. The concept of a CoP is helpful to understand and to support cooperation, knowledge management and collaborative learning [BrDu00]. The CoP can be seen as shared histories of learning [Weng98]. CoP combines the social practice of the community and the identity of its individual members. Moreover, the term professional communities refers to communities of practice in some professional domains, e.g. medicine or construction, where the professional inherently learns by social participation in CoP. In organizations, informal communities of practice want to share knowledge about their professions.

Information systems for professional communities face several challenges. Principles like legitimate peripheral participation, group knowledge, situated learning, informality and co-location need to be considered in the design of the information system. First, community membership and social status are highly dynamic and vaguely defined. The number of users to support with a community information systems can oscillate between tens of users to millions of users in short time frames. Second, the development process of information systems is less stable. Commonly, community members act as stakeholders in the requirements engineering which results in the need of continuous and collaborative information system adaptation.

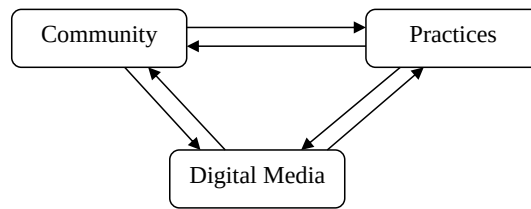


Figure 2.11: Information system key components for communities of practice (adapted from [DDJ*98])

For example, the workplace is currently being shifted from centralized offices to mobile on-site places, causing transformation of professional communities into mobile communities. With respect to their IT needs, mobile communities introduce unique requirements. These on-site communities are characterized by a high degree of collaborative work, mobility and integration of data coming from many members, devices and sensors. A mobile community, therefore, needs tools for communication, collaboration, coordination, and sensing as well as for member, community, and event awareness. Community members are often distributed geographically, therefore, their interactions are mediated by digital channels for direct communication and for indirect exchange of information objects. Tools for communication are natively supported on mobile devices. However, multiple forms of communication, such as voice, messages, chat, and video streaming, should be supported too.

Multimedia information systems have played a significant role in supporting professional communities. Members of a mobile community need to collaborate around different multimedia artifacts, such as images or videos. On the data management level, various data needs to be captured, created, stored, managed, and prepared for further processing in applications. However, the massive amounts of user-generated multimedia content does not necessarily imply content quality and social value. Mobile multimedia information systems need to empower individuals and communities with services for adding value to the content easily. For example, techniques from data mining, machine learning, computer vision, and recommender systems can help to further detect/filter/sort/enhance multimedia content.

As shown in Figure 2.11, the development of information systems for communities of practice needs a support for digital media and related communication tools between community members and collaboration tools over digital media objects. For long time, high-quality multimedia was reserved to professional organizations equipped with expensive hardware. The distribution of multimedia was limited to hard copies such as VHS, VCD, and DVD. The development of Web 2.0, inexpensive digital cameras and mobile devices has spurred Internet multimedia rapidly. People can generate, edit, process, retrieve, and distribute multimedia content such as images, video, audio, graphics, etc., much easier than before.

Web 2.0 paved the way of knowledge sharing for communities of practice. Web 2.0 represents concepts and tools that put a more social dimension into operation and foster collective intelligence. As more members use Web 2.0 services, the better they become. Such services allow people to create, organize and share knowledge, but also to collaborate,

Table 2.1: Comparison of key features between the Web 2.0 and cloud computing paradigms

Web 2.0	Cloud Computing
Massive amounts of content	Externalization of computing and storage
Limited practices (sharing, delivery)	Easier to change practices by changing the cloud infrastructure
Pre-defined multimedia operations and business models	Flexible composition of cloud services based on pay-as-you-go (utility) model and Software as a Service (SaaS)

interact virtually and make new knowledge. However, Web 2.0 models do not solve the issue with continuous changes of IS infrastructure for community services. The ultimate goal of community information systems is to remove the IT engineers from the development loop. The communities should be able to monitor, analyze and adapt their information systems independently. Fortunately, the complexity of existing IT systems has resulted in finding new ways for provision and abstraction of systems, networks and services. Cloud computing can be considered as an innovation driver for information technology — its concept imposes new set of requirements on IT systems (scalability, dynamic reconfiguration, multi-tenancy, etc.) which in turn result in technological breakthroughs. Previous to cloud computing, multimedia storage, processing and distribution services were provided by different vendors with their proprietary solutions.

With the rise of Web 2.0, the amounts of data now available to collect, store, manage, analyze and share is growing continuously. It is becoming common that applications need to scale to datasets of the magnitude of the Web or at least to some fraction of it. Tackling large-data problems is not just reserved for large companies but it's being done by many small communities and individuals. However, it is challenging to handle the large amounts of data within own organization without having large budget, time and professional expertise. This is why cloud computing could make techniques for search, mining and analysis easily accessible to anyone from anywhere. Large-scale data and cloud computing are closely linked.

Cloud computing is fundamental to Web 2.0 development — it enables convenient Web access to data and services. Sharing large amounts of content is one of the salient features of Web 2.0. Clouds further extend these capabilities with externalization of the computing and storage resources. Clouds provide virtually “unlimited” capacities at users’ disposal. Cloud computing further extends Web 2.0 with means for flexible composition of arbitrary services, where the user pays only for the consumed resources. This enables communities to be supported with personalized set of services, and not just typical ones, which is important for professional communities with specialized needs. There is no minimum fee and startup cost. Cloud vendors charge go storage and bandwidth. No installation of media software and upgrade are needed. Expensive and complex licenses are aggregated in the cloud offers. The always-on cloud storage provides world wide distribution of popular multimedia content. The cloud can alleviate the computation of user devices and save battery energy of mobile phones. These can greatly facilitate small amateur organizations and individuals. Cloud computing boils down multimedia sharing to simple hyperlink sharing. Sharing through a cloud also improves the QoS because cloud-client connections provide higher bandwidth

and lower latency than client-client connections.

Cloud computing is a factor for change and innovation thanks to the increased efficiency of information technology infrastructure utilization leading to lower costs. Cloud computing reduces the cost effectiveness for the implementation of the hardware, software and license for all. Large data centers benefit from economies of scale. It refers to reductions in unit cost as the size of a facility and the usage levels of other inputs increase [SuSh03]. Large data centers can be run more cost efficiently than private computing infrastructures. They provide resources for a large number of users. They are able to amortize the demand fluctuations per user basis, since the cloud resources providers can aggregate the overall demand in a smooth and predictable manner.

2.6.1 Media-centric Theory

In order to successfully support the practices of any kind of a professional community, independent of the size or domain of interest, understanding the knowledge sharing processes within communities is needed. Supporting professional community practices faces many challenges. There are several reasons that perplex the implementation of successful community IS, namely processes like situated learning, shared group knowledge, mobility and co-location need to be taken into account when designing the information system. Moreover, the needs are community-specific. Community members are not able to express precise requirements at the beginning; i.e. the requirements emerge along the system use. Therefore, the community needs mechanisms to add, configure and remove services on the fly. Besides, the advances of multimedia technology requires constant support of novel hardware and network capabilities. A full spectrum of multimedia content technologies needs to be supported.

The central process in professional communities is sharing of knowledge about the profession. Organizational knowledge management and professional learning are closely connected. The socialization, externalization, combination and internationalization (SECI) model by Nonaka and Takeuchi [NoTa95] has been widely accepted as a standard model for organizational knowledge creation. It emphasizes that knowledge is continuously embedded, recreated and reconstructed through interactive, dynamic and social networking activity. Spaniol et al. [SKCa09] further refined the SECI model as a media-centric knowledge management model for professional communities. It combines the types of knowledge of community members, tacit and explicit knowledge, and the process of digital media discourses within communities of practice and their media operations.

The media-specific theory [FoSc04] distinguishes three basic media operations:

- Transcription: a media-specific operation which makes media collections more readable
- Localization: an operation to transfer global media into local practices. It can be further divided to:

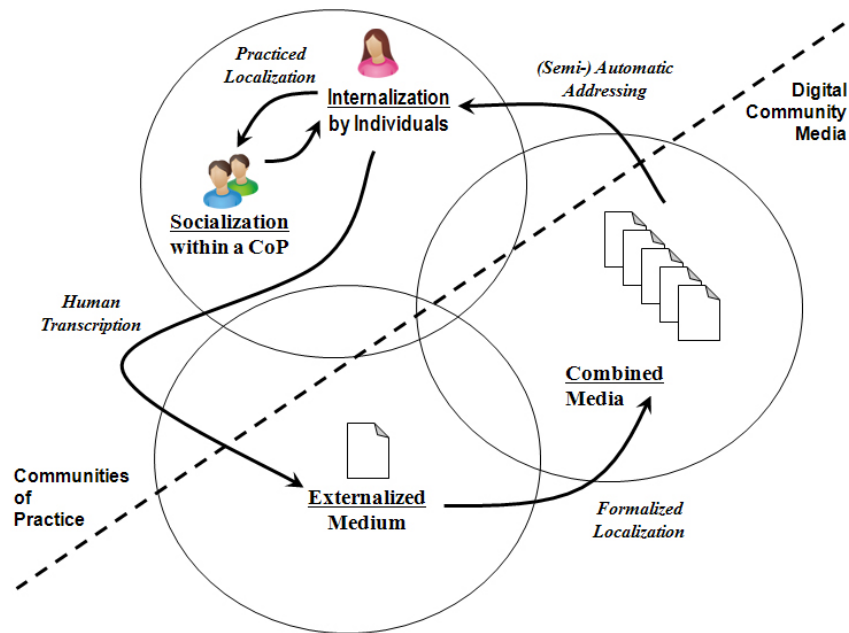


Figure 2.12: Media centric theory of learning in communities of practice (adapted from [SKCa09])

- Formalized localization
- Practiced localization
- (Re-)addressing: an operation that stabilizes and optimizes the accessibility in global communication.

Spaniol et al. [SKCa09] integrate these media operations into the learning and knowledge sharing processes of professional communities (see Figure 2.12). As seen in the figure, the individuals internalize knowledge from some sources. The knowledge is then communicated with others in two different ways. One way is the human-human interaction which is called practiced localization which, in turn, fosters content’s socialization within the CoP thus forming a shared history. A second way is by human transcription which means creating new digital artifacts on an externalized medium. The externalized artifacts are then processed by the information system which is called formalized localization of the media artifacts. The artifacts are combined and made available for further use. The semi-automatic addressing operation closes the circle which represents a context-aware delivery and presentation of the medial artifacts.

Table 3 gives an example mapping between the media operations defined in the theory and possible cloud services. This mapping is not complete nor extensive. Many of these services benefit from the cloud infrastructure in terms of computing, storage and networking resources, e.g. multimedia transcoding and recommender systems.

Table 2.2: Mapping between media-theoretic operations and cloud services

Media-theoretic operation	Cloud multimedia service
Transcription	• Metadata creation • Ubiquitous multimedia acquisition with digital media devices • Physical-to-virtual input methods: OCR, object recognition, voice recognition
Formalized localization	• Real-time audio/video/text communication • Multimedia transcoding • Multimedia indexing and processing • Story creation
Practiced localization	• Content and metadata collaboration • Multimedia sharing • Tagging • Storytelling
Re-addressing	• Recommender Systems • Multimedia retargeting • Adaptive streaming • Mobile augmented reality

2.7 Summary

The contribution of this chapter lies in the introduction of technological and conceptual underpinnings of cloud computing. The cloud paradigm envisions optimizations in information technology (lower costs, dynamic computing environments), and by which it fosters new innovation developments. Cloud computing has many characteristics such as elastic, utility-like, pay-as-you-go resource usage. These cloud principles can also be applied to the resource-constrained, network-volatile mobile environments. The benefits are manifold, e.g. augmented computing and storage capacities for mobile devices, extended battery duration, provided ability to consume resource-demanding applications, etc. Beside that mobile service backends can benefit enormously from cloud infrastructures, other cloud computing solutions do not lend themselves to be applied in mobile cloud settings. In fact, to achieve cloud benefits, approaches like adaptive remote execution, cyber-foraging, cross-device synchronization and ad-hoc computing gain in importance for MCC.

While most of the tools using cloud computing deal with enterprise or big data applications, in this research work we have focused on applying cloud technology to multimedia applications. Multimedia applications have unique requirements and they need to be matched with the cloud approach. This has two sides, namely, cloud services that are designed and optimized for (mobile) media, and multimedia formats and data structures that are aware of the cloud infrastructure. These issues are discussed in Chapter 4.

Another aim of this dissertation is to shed light on how mobile cloud computing can be applied to support the practices of professional communities, especially, their needs from point of view of multimedia and knowledge sharing information systems. Mobile professional communities exhibit a complex structure and dynamic processes which is reflected on the information system support. The utility-like resource provisioning of cloud resources suits perfectly to the dynamic membership nature of professional communities. Software as a Service (SaaS) concept of cloud computing provides means for unlimited configurations and mashups of community services to match any emerging IS requirement. Chapter 6 lists several prototypes that confirm the benefits from applying cloud services for the professional communities support.

Science is organized knowledge.
Wisdom is organized life.

Immanuel Kant (1724 - 1804)

Chapter 3

State of the Art

This chapter presents a review of the state of the art regarding approaches, tools and infrastructures that support the building of mobile multimedia cloud services and applications. This dissertation is positioned at the intersection between three domains, namely cloud, mobile and multimedia computing (see Figure 3.1). The latter two research domains have been explored for more than two decades. However, the paradigm shift caused by cloud computing also raises unique issues and challenges in those domains. This chapter presents relevant aspects from each domain that can be applied to the other domains. Section 3.1 presents techniques for processing of multimedia which can be performed on a distributed cloud infrastructure. In addition, in the same section, cloud platforms specialized for multimedia handling are included, too. How cloud principles can be applied in mobile settings and how the mobile environments shape mobile cloud computing are discussed in Section 3.2. Section 3.3 conducts a survey of selected aspects of particular relevancy for mobile multimedia. The insights provided in this chapter form the basis of derivation and identification of a conceptual model and key requirements which are described in the following chapter.

3.1 Cloud-based Multimedia Systems

Recently, several research papers have reported about applying the cloud computing infrastructures and platforms for solving challenging multimedia processing problems in different domains. For example, Schmidt and Rella [ScRe11] have presented an approach for processing large and non-uniform media objects on a MapReduce-based cluster. White et al. [WYLD10] presented Web-scale computer vision algorithms using MapReduce for multimedia data mining. Summa et al. [SVPS11] reported on application for massive image editing with Hadoop [Hado09].

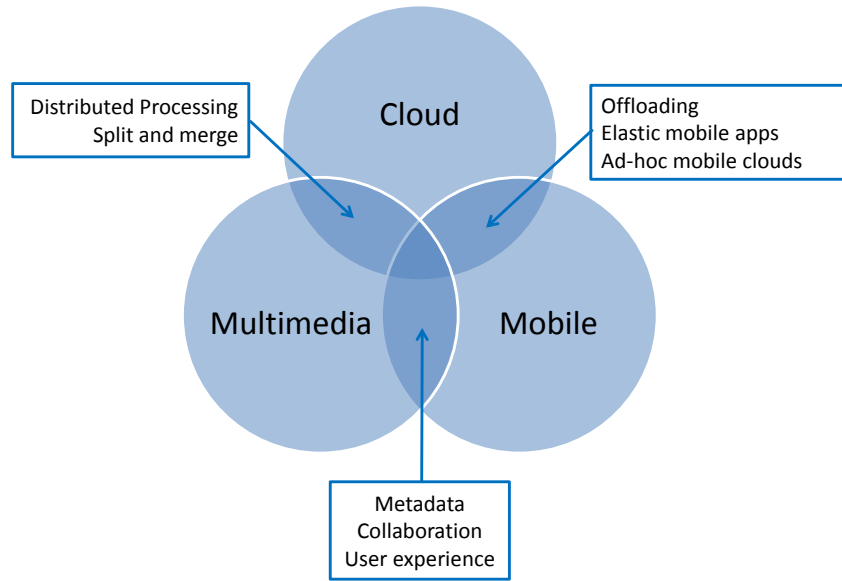


Figure 3.1: Related work at the intersection of the main research areas

3.1.1 Multimedia Processing in the Cloud

Media content distribution to end-user devices is challenging due to the many media formats. The processing of media content to support multiple client types since it is a very CPU-intensive task that requires long-processing times while causing high latency. Cloud computing can decrease the time at which content is transcoded and made available to the end user.

x264farm [x26406] is one of the first approaches to exploit distributed video encoding using a cluster of PCs in a local network. It uses a free H.264/AVC encoder to encode large video files into the H.264 format. The architecture consists of agent components executing on slave machines and a controller component running on a master machine. A video file that needs to be transcoded is transferred entirely between compute nodes which induces excess network traffic. The compute cluster has static configurations without any load balancing. This approach is, therefore, limited for cloud purposes. Nevertheless, x264farm demonstrates the feasibility of harvesting the power of distributed computing environments composed of commodity hardware, i.e. PCs.

A good introductory example for video processing in the cloud is VideoToon [Sand11]. Its main purpose is to aid transform process of images and videos into cartoons. To realize the transformation, it uses Hadoop and Intel VT Virtualization Technology. First the

configuration of the system including selection of a pricing model is specified. The system then creates the virtual cluster and configures Hadoop. This procedure takes less than five minutes. The video processing logic itself uses MapReduce jobs. The first one splits the stream into sub-streams. The second one transforms the sub-stream and the last one joins the transformed streams in the correct order. After the join phase it is possible to download and watch the resulting video stream. Similarly, splitting a video stream into sub-streams or sub-chunks is also used by other researchers. Garcia et al. [GKFu10] present an approach where, again, a Hadoop-based cloud is used as an aid the transcoding of media content for an HTTP Live Streaming scenario. Different devices and different Internet connections require various video formats. But saving all possible video formats produces high costs and is inflexible to user-specific needs. Three different variations of video formats are produced. To achieve that, the video is split into the same number of pieces — content-based or fixed duration segments — and the parts are distributed among MapReduce nodes. They showed that the delay between a request and a retrieval of a video is shortest when the video is split into the same number of pieces as available nodes.

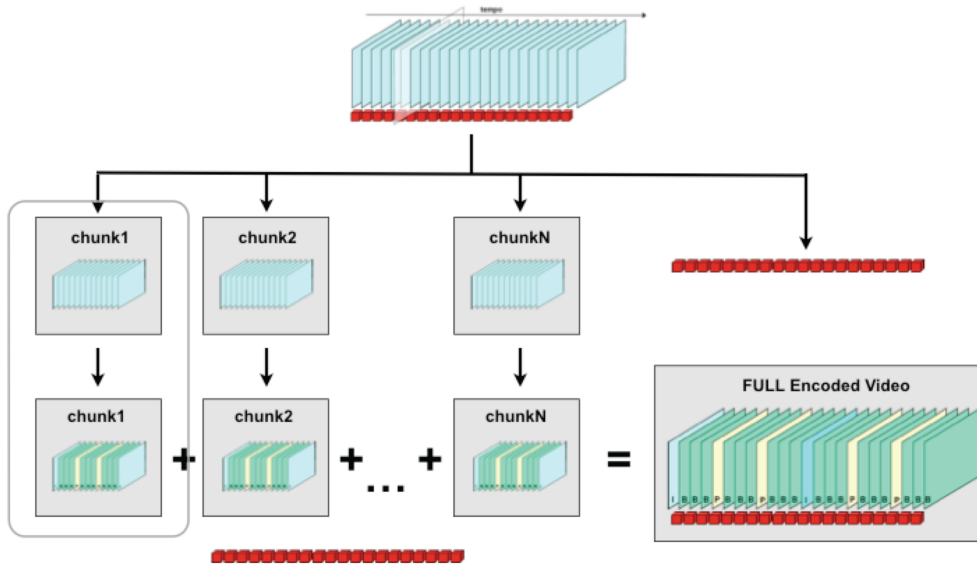


Figure 3.2: The Split&Merge approach of video encoding (adapted from [PABE10])

Pereira et al. [PABE10] also propose MapReduce workflow for video transcoding applications. Their approach, shown on Fig. 3.2, is based on splitting of the video into chunks, and then processing the chunks in parallel using MapReduce jobs. Instead of splitting videos in sequences of a predefined duration, they propose content based analysis to be performed first. The analysis is important as some parts of the video may be compressed already. Therefore, key frames have to be extracted and these key frames then are the main parameters for the chunk separation.

All these approaches demonstrate the advantages of using MapReduce for video processing, e.g. parallelization, which result in much faster video processing. Furthermore, Intel

Research proposed the use of MapReduce to speed up image processing in special feature detection. This showed that MapReduce is not limited to speedup of transcoding, but can be used for feature detection in videos where frames can be used as images [ChSc08].

Another approach in video processing using the cloud is the combination of cloud computing and peer-to-peer (P2P) technology. Fouquet et al.[FNCa09] propose a system in which a cloud-based infrastructure is integrated into a P2P system. The hybrid system for video streaming service tries to alleviate several issues with P2P systems. For example, lack of enough peers or various upstream capability of the peers, can affect the network. and, thus, the end-user video experience. In such cases, Fouquet et al. propose to augment with a cloud infrastructure. The system asks if any peer wants to pay for better quality. If there is someone responds positively, the system will configure a virtual machine and add it to the peers to improve the streaming quality. Nevertheless, it is questionable if such a system will be accepted and understood by the users.

Schmidt and Rella [ScRe11] have presented an approach and its implementation that utilizes the MapReduce model for the processing of audio-visual content. This system is capable of analyzing and modifying large audiovisual files using multiple computer nodes in parallel. They also discuss the programming model and its application to binary data. Figure 3.3 shows a MapReduce application for processing a set of video files (AV1, AV2, AV3). The files are available on a Hadoop distributed file system which provides a shared storage network across the work nodes (nodes 1-3). It is important to provide suitable mechanisms to divide large audiovisual files into parts to support the parallel processing of binary input data. However, most binary formats cannot be split or do not support division at all. For processing of small size files this problem can be solved by splitting the payload on a per-file basis. When the size of audio-visual content is large and it needs too much resources on a single processor architecture, a promising solution is to use multiple nodes for a single video file with parallel processing to speed up the execution time. In order to divide a media byte-stream into parts, it is important to identify key frame positions in the media container. Schmidt and Rella's extension to Hadoop MapReduce framework allows to process heterogeneous collections of media content directly from the storage. They report almost a linear speedup which reduces response times by increasing working nodes in the test case. However, this framework is only a partial solution since it provides mechanisms for generating video records as input values that can be passed to a Map function, but it does not provide the required abstractions and mechanisms to analyze and transform the video record. In fact, many media operations require that certain algorithmic operations, such as computer vision techniques, to be applied on the video records.

White et al. [WYLD10] explored computer vision algorithms using MapReduce framework that are relevant to the data mining community. They compared several computer vision algorithms with implementation aspects, e.g. background subtraction, classifier training, clustering, sliding windows, bag-of-features and image registration. They also reported experimental results on a 410 node Hadoop cluster for k-means and single Gaussian background subtraction. The Hadoop implementation of MapReduce has an input format named

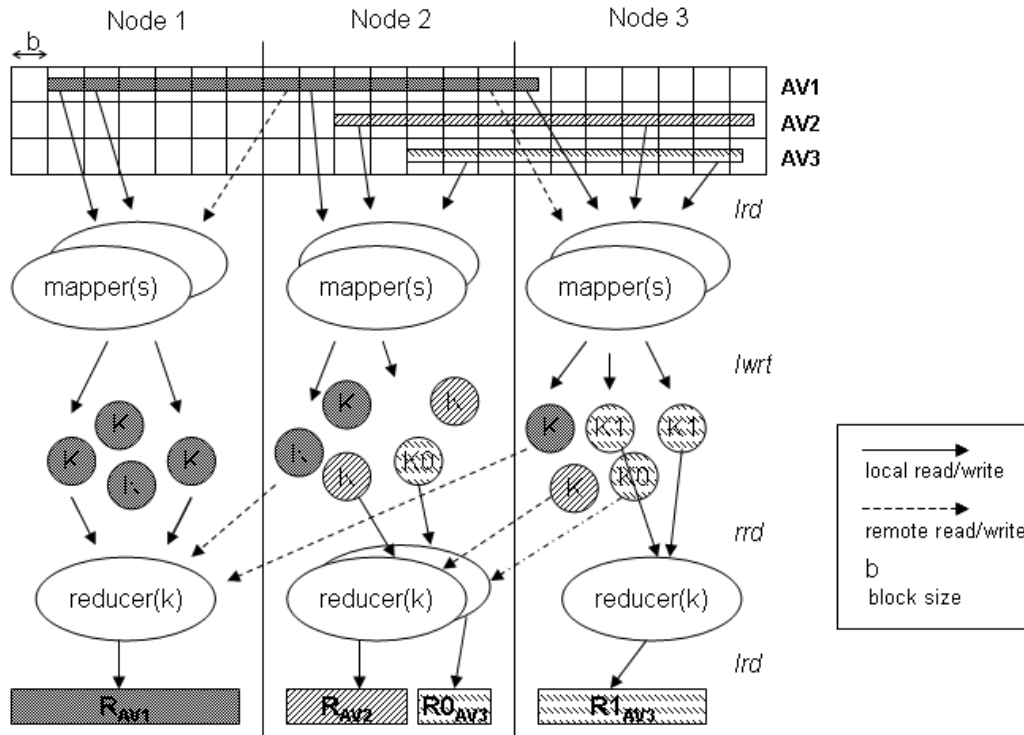


Figure 3.3: A data flow for processing audiovisual data using the MapReduce model (adapted from [ScRe11])

Sequence-Files that allows to represent images and arrays in original binary which lets it parse efficiently, reduces space and computation requirements.

As described here, we can see that cloud principles can be used for various tasks of video and multimedia processing. The first research attempts to apply these principles were driven by success stories of cloud-based document and text processing applications. MapReduce is the common framework to tame the power of cloud infrastructure. However, several modifications are needed to apply it on multimedia data, i.e. to be able to read/write and compress/decompress media formats and separate files at certain positions.

3.1.2 Cloud-aware Multimedia

The term *cloud-aware multimedia* stands for multimedia formats and protocols that let multimedia services easily take advantage of cloud resources and also enable impoverished mobile devices be used in complex tasks like video editing directly on the device. Cloud-aware multimedia provides means to share work processes and data between a device and a cloud infrastructure. The multimedia content is optimized for storage and processing on cloud infrastructure, and also optimized for delivery over volatile mobile networks.

A common problem for mobile streaming and online videos are the large single video files. ChunkStream [PaPh10] is a system to overcome this problem for efficient streaming and

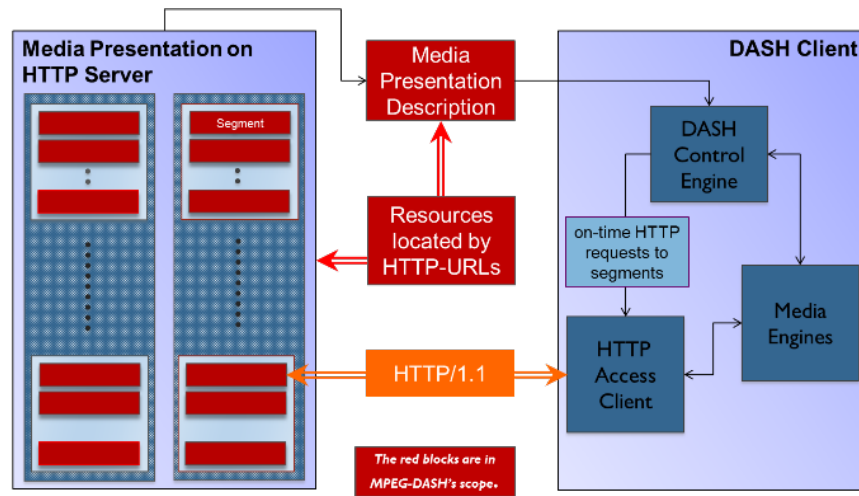


Figure 3.4: Overview of a MPEG-DASH streaming system (adapted from [Stoc11])

editing of online videos. In contrast to using specialized protocols or streaming formats, it uses file *chunks*. Chunks are size-fixed arrays that contain scalar data and references to other chunks. It allows to expose large data structures over the network. ChunkStream itself represents video clips using chunks. This enables to retrieve and work with only the part of the video which is needed. It supports resource-adaptive playback and live streaming. ChunkStream embeds video chunks into general data structures like search trees, which enables quick finding of video frames, frame-accurate editing of chunks and adaptive streaming of video. Composite data structures can be created out of the chunks using links to other chunks, which can even reside on different computers. In this way, a device can only work with portions of the clips that it needs which results in bandwidth-efficient video streaming and efficient editing mechanism for remote video.

The industry has seen the benefits of cloud-aware multimedia and therefore, tries to adopt an inter-operable standard such as the DASH standard [Stoc11] which has a similar approach of representing multimedia content as ChunkStream. In DASH, the content is stored as segments that can fit in single HTTP GET response to the requesting clients (see Figure 3.4). Each media segment is addressable separately by a unique URL. Multiple segments constitute a media stream which can be delivered over HTTP to any client. There are many benefits from this approach. Clients can automatically switch between video and audio streams, thus adapt to the network conditions and user preferences. The same content can be accessible from multiple locations, i.e. URLs, thus enabling reuse of wide-spread HTTP caching infrastructure.

Another exemplification of cloud-aware media are *gigapixel images*. It is a digital image bitmap composed of one billion (10^9) pixels, more than 1000 times the information captured by an one megapixel digital camera. Gigapixel technology is being deployed in a wide range of applications ranging from remote sensing to the field of arts. Current technology for creating such very high-resolution images involves making mosaics of a large number

of high-resolution digital photographs. As the gigapixel images popularity increases, cloud resources as a commodity offer are the best opportunity to adapt high level of abstraction and extend the distribution to a much wider community. But how to extend a particularly expensive technique like gradient domain image processing (seamless cloning, panorama stitching and high dynamic range tone mapping) to the cloud is a big challenge.

Summa et al.[SVPS11] provided a solution for massive image editing on top of Hadoop. They presented a new tiling method to solve a Poisson system for large scale image editing with small memory and disk storage footprints. They also gave a practical example of extending graphics gradient domain algorithms to use the MapReduce framework and address details for efficient implementation. Seamless cloning, panorama stitching and high dynamic range tone mapping belong to gradient domain techniques. Gradient based techniques manipulate an image based on the value of gradient field. In practice, large images like panoramas are stored as a set of tiles. The processing needs two phases. First phase is to up sample a precomputed coarse solution and solve each tile to produce a smooth solution. Second phase makes a smooth image of the tiles that significantly overlap the smoothed tiles from the first phase. After loading/combining the partial tiles, a Map procedure runs an iterative solver initialized with the up-sampled pixel color. Each mapper emits a key/value pair, where values as small as tile data and key are computed as a row/column pair in the space of the large tiles. The Reduce phase then gathers the n smaller tiles that make up the overlapping window. After data gathering, the gradients are computed from the original pixel values and an iterative SOR solver is being run. Saving individual tiles is not efficient in Hadoop HDFS, therefore, data is saved as large tiles comprised of four small tiles, which can be used as mapper's input in the first phase.

Whereas traditional multimedia formats consider storage of multimedia data on a single computer (server), cloud-aware multimedia embraces the distributed nature of cloud infrastructure. Moreover, novel multimedia delivery protocols integrate with the widespread of the HTTP protocol and benefit from the cloud-aware multimedia formats. They can adapt the quality of services (and thus user experience) contingent to the services demands and network status.

3.2 Mobile Cloud Computing

The fundamental challenges in mobile computing have already been summarized by several researches [Saty96, Gupt08, JHE199]. Mobile computing environments are characterized by severe resources constraints and frequent changes in operating conditions. Mobile devices inherently have and will continue to have limited resources as processing power, memory capacity, display size, and input forms. These have been the forming factors of existing mobile application approaches.

3.2.1 Traditional Mobile Computing Models

In the traditional mobile computing, applications and services fall into two main categories, i.e. offline and online applications. This categorization emerged mainly as the distributed and Web computing models were translated for mobile settings with the advent of mobile technology. To leverage the full potential of mobile cloud computing we need to consider the capabilities and constraints of existing architectures.

Offline Applications

Typical applications available for modern mobile devices fall into this category. They act as fat client that processes the presentation and business logic layer locally on mobile devices with data downloaded from backend systems. There is a periodical synchronization between the client and the backend system. A fat client is a networked application with most resources available locally, rather than distributed over a network as is the case with a thin client.

Offline applications, also often called native applications, offer:

- good integration with device functionality and access to its features
- performance optimized for specific hardware and multitasking
- always available capabilities, even without network connectivity

On the other hand, these native applications have several disadvantages:

- no portability to other platforms
- complex code
- increased time to market
- a requirement for developers to learn new programming languages

Online Applications

An online application assumes that the connection between mobile devices and backend systems is available most of the time. Smartphones are popular due to the multitude of applications, but there are problems such as cross-platform issues. Here, Web technologies can overcome them; applications based on Web technology are a powerful alternative to native applications.

Mobile Web applications have the potential to overcome some of the disadvantages of offline applications because they are:

- multi-platform

- directly accessible from anywhere
- knowledge of Web technologies is widespread among developers, greatly minimizing the learning curve required to start creating mobile applications

However, mobile Web applications have disadvantages:

- too much introduced latency for real-time responsiveness, (even 30 msec latency affects interactive performance [SBCD09])
- no access to device's features such as camera or motion detection
- difficulties in handling complex scenarios that require keeping communication session over a longer period of time

More opportunities for mobile online or web applications opens the new upcoming HTML 5 standard [HTML5]. It will allow offline execution support through cache management and database storage, more multimedia features, geolocation API and more. The mobile web applications are currently the most promising option for building a cross-device mobile ecosystem. HTML 5 is a critical piece for the mobile Web.

Issues with Offline and Online Mobile Applications

Current applications are statically partitioned, i.e. most of the execution happens on the device or on backend systems. However, mobile clients could face wide variations and rapid changes in network conditions and local resource availability when accessing remote data and services. As a result, one partitioning model does not satisfy all application types and devices. In order to enable applications and systems to continue to operate in such dynamic environments, mobile cloud applications must react with dynamical adjusting of the computing functionality between the mobile device and cloud depending on circumstances. In other words, the computation of clients and cloud has to be adaptive in response to the changes in mobile environments [GNM*03].

3.2.2 Application Models for Mobile Cloud Computing

Mobile cloud computing could be described as the availability of cloud computing services in a mobile ecosystem, i.e. world wide distributed storage system, ability to exceed traditional mobile device capabilities and to offload processing, storage and security.

Augmented Execution

The simplest way to augment weak devices such as mobile phones is the application delivery based on the traditional client-server model [JHEI99]. However, the client-server model does not consider the changing conditions in pervasive computing environments, causing limited interactivity (thin clients, Web applications) or less portability (fat clients, native mobile

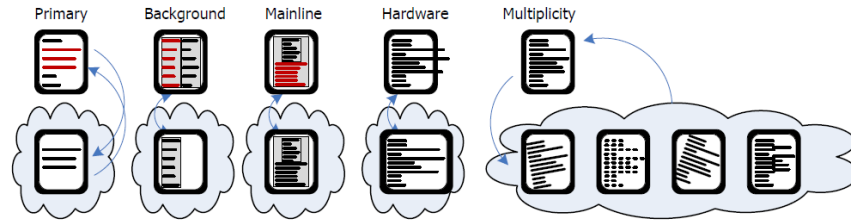


Figure 3.5: CloneCloud categories for augmented execution (adapted from [ChMa09])

applications) [WHMa10]. Several projects in the beginning of mobile computing tried to leverage these issues. In the Spectra project [FNSa01], programmers define execution plans that run several application partitioning variants which deliver different quality of service. The Coign project [HuSc99] is a nice example of automatic partitioning of Microsoft DCOM applications [DCOM] without source code modification, it outputs client-server applications again statically.

Augmented execution refers to a technique used to overcome the limitations of smartphones in terms of computation, memory and battery. Chun and Maniatis [ChMa09] proposed an architecture that addresses these challenges via seamlessly offloading execution from the phone to computational infrastructure (cloud) where a *cloned replica* of the smartphone's software is running. More recently, Kosta et al. [KAH*11] have further improved this idea. Although such virtualized offloading can be considered as being a simple and general solution, it lacks flexibility and control over offloadable components. Therefore, we consider that application developers can better organize their application logic using the established Android OS service design patterns.

The mobile phone hosts its computation and memory demanding applications. However, some or all of the tasks are offloaded in the cloud where a cloned system image of the device is running. The results from the augmented execution are reintegrated upon completion. This approach for off-loading intensive computations employs loosely synchronized virtualized or emulated replicas of the mobile device in the cloud. Thus, it emulates illusions that the mobile user has a more powerful, feature-rich device than actually in reality, and that the application developer is programming such a powerful device without having to manually partition the application or provision proxies. Instantiating device's replica in the cloud is determined based on the cost policies which try to optimize execution time, energy consumption, monetary cost and security.

Figure 3.5 shows a categorization of possible augmented execution for mobile phones: (1) primary functionality outsourcing - more like a client-server application, (2) background augmentation - good for independent separate process that can run in background like a virus scanning, (3) mainline - in-between primary and background augmentation, (4) hardware - the replica runs on a more powerful emulated VM, and (5) multiplicity - helpful for parallel executions.

Another similar approach of using virtual machine (VM) technologies that wraps the

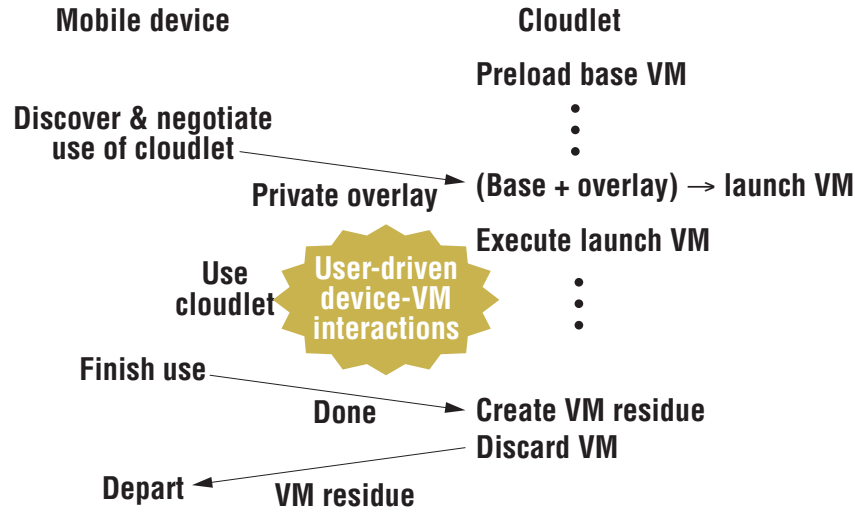


Figure 3.6: Dynamic virtual machine synthesis timeline (adapted from [SBCD09])

execution of compute-intensive software from mobile device is presented by Satyanarayanan et al. [SBCD09]. In their architecture, a mobile user exploits VMs to rapidly instantiate customized service software on a nearby cloudlet and uses the service over WLAN. A *cloudlet* is a trusted, resource-rich computer or a cluster of computers well connected to the Internet and available for use by nearby mobile devices. Rather relying on a distant cloud, the cloudlets eliminate the long latency introduced by wide-area networks for accessing the cloud resources. As a result, the responsiveness and interactivity on the device are increased by low-latency, one-hop, high-bandwidth wireless access to the cloudlet. The mobile client acts as thin client, with all significant computation occurring in a nearby cloudlet. This approach relies on a technique called dynamic VM synthesis (see Fig. 3.6). A mobile device delivers small VMs overlay to the cloudlet infrastructure that already owns the base VM from which this overlay was derived. The infrastructure applies the overlay to the base to derive the VM which starts executing in the precise state in which it was suspended. However, Satyanarayanan et al. [SBCD09] report that the VM synthesis takes 60 to 90 seconds, which might not be acceptable for performing simple or ad hoc tasks. Garriss et al. [GCB*08] use a similar principle of running own VMs on public kiosks in order to establish a trustworthy and personalized computing environment. The user leverages a personal mobile device to gain degree of trust in a kiosk prior to using the kiosk. Using VMs enables the user to resume a complete personal computing environment that includes own choices of operating system, applications, settings, and data.

Elastic Applications

Running of applications in heterogeneous volatile environments such as mobile clouds requires dynamic partitioning of applications and remote execution of some components. Applications can improve their performance by delegating part of the application to remote execution on a resource-rich cloud infrastructure.

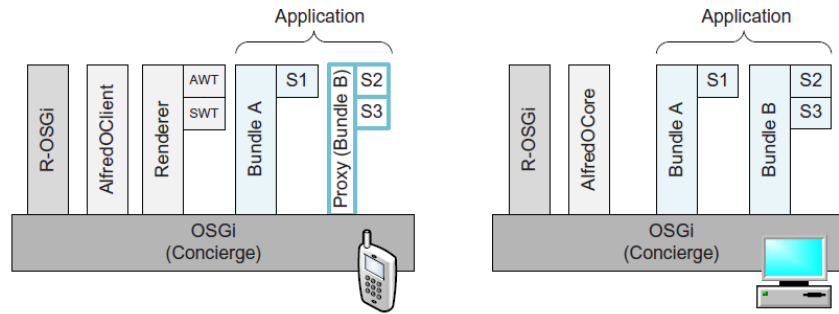


Figure 3.7: AlfredO modular architecture based in OSGi (adapted from [GRJ*09])

Giurgiu et al. [GRJ*09] developed an application middleware that can automatically distribute different layers of an application between the device and the server while optimizing several parameters such as latency, data transfer, cost, etc. In the core of this approach is a distributed module management which automatically and dynamically determines when and which application modules should be offloaded, in order to achieve the optimal performance or the minimal cost of the overall application. Giuriu et al. use the AlfredO [RRA108] framework to carry out the distribution of the application modules between the mobile phone and the server. The AlfredO framework allows developers to decompose and distribute the presentation and logic layer of the application, while the data layer always stays on the server side. The minimal requirement is the UI of the application to run on the client side. Furthermore, Rellermeyer et al. [RDA109] showed how such a modular application model enables elasticity. *Elasticity* in software can be observed as the ability to acquire and release resources on demand. Modules are units of encapsulation and units of deployment that compose the distributed application. The underlying runtime module management platform hides most of the complexity of the distributed deployment, execution, and maintenance.

AlfredO is based on R-OSGi [RARo07], a conceptual extension of the OSGi middleware model, that allows decomposition of Java applications in *software modules*. A modified version of the original OSGi, namely R-OSGi, is used because the original OSGi allows only running services on the same Java virtual machine. Figure 3.7 shows the main concept. After the connection is established, the client requests an application. Then the optimal deployment for the application is computed. Based on that decision, an application description and a list of services to be fetched are sent to the client's Renderer. The Renderer generates a corresponding UI according to the description. Furthermore, for the services that are decided to run on the client side the corresponding service bundles are fetched (see Fig. 3.7 service S1). Otherwise, for the services that are decided to run on the server, a local proxy on the client is created as an interface to this services (services S2 and S3).

Similarly, MAUI [CBC*10] is a system that enables fine-grained offload of mobile code to the cloud infrastructure. MAUI's goal is to maximize battery life of device with code offload. Developers annotate while programming which methods can be offloaded for remote execution. The profiling information for once offloaded methods is gathered, which is later

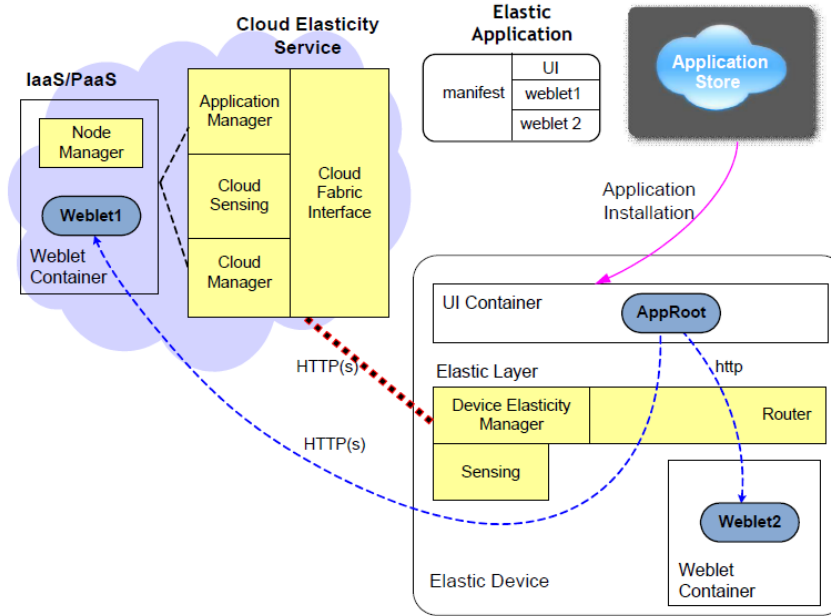


Figure 3.8: Reference architecture for elastic applications (adapted from [ZJKG10])

used to better predict future invocations whether methods should be offloaded. The profiling information, network connectivity measurements, bandwidth and latency estimations are used as input parameters for an optimization problem which is periodically solved to give a decision which methods and when should be offloaded. Compared with [GRJ*09], MAUI allows a fine grained offloading mechanism on the level of single methods, where in [GRJ*09] the offloading happens on complete software modules. Even the experimental results from MAUI show that the separate method offloading can be contra-productive, i.e. several methods should be combined to achieve benefits.

Zhang et al. [ZSG*09, ZJKG10] develop a reference framework for partitioning a single application into elastic components with dynamic configuration of execution. The components, called *weblets*, are platform independent and can be executed transparently on different computing infrastructures including mobile devices or IaaS cloud providers such as Amazon EC2 and S3 [AWS]. The application is split down to a UI component, weblets, and a manifest describing the application (see Fig. 3.8). Weblets are autonomous functional software entities that run on the device or cloud, performing computing, storing and network tasks. An elasticity manager component decides on migration, instantiation and migration of the weblets. This processes are transparent to the running application. The advantage of using such independent functional units — weblets — over AlfredO and R-OSGi is that weblets are not tied to one particular programming language or specification, allowing a wider range of possible applications.

Ou et al. [OYZh07] propose a class instrumenting technique, i.e. a process to transform the code classes into a form which is suitable for remote execution. Two new classes

are generated from the original class, one is an instrumented class which has the real implementation and the same functionality as the original class, the other is a proxy class, whose responsibility is only to call the function written in the instrumented class. Then, the instrumented class can be offloaded to remote cloud, and the call will be invoked from the instrumented proxy in the remote cloud. In MACS (see Section 5.3.2), we adopt a similar idea, but unlike Ou et al. [OYZh07] we use a standardized language for the proxy interfaces which is already widespread in the Android platform. The Cuckoo framework [KPKB10] and MAUI system in [CBC*10] implement a similar idea. Our MACS middleware is inspired by these solutions. However, MACS middleware does extra profiling and resource monitoring of applications and adapts the partitioning decision at runtime.

An important challenge in partitioned elastic applications is how to determine which parts of code should be pushed to the remote clouds. The graph based approach to model the application has been used in several works. Giurciu et al. [GRJ*09] use “consumption” graph and decide which part should be running locally or remotely by finding a cut of the consumption graph with a goal function, which minimizes the total sum of communication cost, transmitting cost and the cost of building local proxies. The AIDE platform [GMG*04] uses a component-based offloading algorithm, which mainly focuses on minimum historical transmission between two partitions. The $(k + 1)$ partitioning algorithm, introduced by Ou et al. [OYZh07], is applied to a multi-cost graph to represent the class-based components. A similar approach is done by Gu et al. [GMG*04, GNM*03]. Zhang et al. [ZSG*09, ZJKG10] use a general Bayesian inference to make the partitioning decision. However, constant execution of graph-cut algorithms or inference algorithms on the mobile device takes significant resources on the constrained device. We use an integer linear optimization model to describe the offloading so that it is not only easy to implement, but it can also be independently solved if the remote clouds are temporarily not available.

Application Mobility

A mobile cloud can be accessed through heterogeneous devices. To provide a seamless user experience, same applications need to run on different devices. *Application mobility* plays a crucial role in enabling the next generation mobile applications. Application mobility is the act of moving application between hosts during their execution. Basically, application mobility is migrating running application states from one device to another to which the user has an immediate access [AMJ*09, KGNi04].

Application mobility is closely related to process migration. Process migration is an operating system capability that allows a running process to be paused, relocated to another machine, and continued there. It represents *seamless mobility* at the granularity of individual processes, and has been the research focus of many experimental projects [MDP*00]. However, application mobility involves more than process migration, e.g. migration tasks to different architectures or UI adaptation.

Satyanarayanan et al. [SKHH05] employed a mechanism called Internet Suspend/Resume (ISR), which allows one to logically suspend a machine at one Internet site, travel to some other sites and then seamlessly resume work there on another machine. ISR implementation

is built on top of the VM technology and a distributed file system. Each VM encapsulates distinct execution and user customization state. The distributed file system transports that state. However, one drawback is that migrating a complete virtual machine consumes more time and bandwidth than just selective application migration. Another drawback is that this works only on one platform type, otherwise the latency is too high. In contrast to ISR, David et al. [DDC*07] proposed an adaptive application mobility solution based on Java-based platform that supports mobile agents across heterogeneous hardware. In this approach, their design solution migrates individual applications and supports adaptation.

Ad-hoc Mobile Clouds

An ad-hoc computing cloud represents a group of mobile devices that serve as a cloud computing provider by exposing their computing resources to other mobile devices. This type of mobile cloud computing becomes more interesting in situations with no or weak connections to the Internet and large cloud providers. Offloading to nearby mobile devices reduces monetary cost, because data charging is avoided, especially favored in roaming situations. Moreover, it allows creating computing communities in which users can collaboratively execute shared tasks.

Huerta-Canepa and Lee [HuLe10] present guidelines for a framework to create virtual mobile cloud computing providers. This framework mimics a traditional cloud provider using nearby mobile devices. The proposed approach allows avoiding a connection to infrastructure-based cloud providers while bringing benefits of computation offloading. However, such an approach requires the support for spontaneous interaction networking with discovery and selection of mobile peers. Hadoop¹ ported on mobile device is used for distributing of processing tasks and storage. Communication is based on XMPP. The Hyrax project [Mari09] employs a similar approach of using the Hadoop framework on mobile devices to share data and computation. Hadoop implements much of the core functionality needed for ad-hoc clouds, including global data access, distributed data processing, scalability, fault-tolerance, hardware interoperability and data-local computation. Since Hadoop is mainly designed for deployment on many servers, the major problem is how to enable the Hadoop framework to run on a mobile device.

Cao et al. [CJK*09] presented a middleware that allows access from mobile devices to a bundle of multimedia services exposed from other mobile nodes. Mobile nodes can host Web services that are accessed by other mobile nodes, thus exposing their computing capacities to the other mobile peers in an ad-hoc cloud. Particularly, related to ad-hoc clouds, much research in mobile ad-hoc and sensor networks has been done up to date, and it is out of the scope of this dissertation.

¹<http://hadoop.apache.org>

3.2.3 Comparison of Mobile Cloud Application Models

A comparison of existing approaches for mobile cloud computing may point out the way to a better solution for mobile applications. The aforementioned application models fulfill in different scales the vision of mobile cloud computing. We have compared the models according to:

- *Middleware*: The enabling underlying technology used to achieve desired system properties.
- *Cost Model*: Are the different parameters of mobile clouds used to provide best performance?
- *Programming Abstraction*: How powerful are the used programming tools to achieve quicker solid applications, while preserving the control over different mobile cloud parts?
- *Solution Generality*: Does the solution work for all applications or only for a few?
- *Implementation Complexity*: How difficult is it to develop mobile cloud applications?
- *Static & Dynamic Adaptation*: What is the separation of responsibilities between mobile clients and the cloud?
- *Network Load*: How large is the volume of data transferred? What is the introduced latency by offloading?
- *Scalability*: Can the application scale?

Table 3.1 shows how each of the approaches maps to the above attributes. The approaches from Cuervo et al. [CBC*10] and Zhang et al. [ZJKG10] received top scores, because their model incorporates a cost model for deciding best execution configuration, the execution can also adapt dynamically. They provide a SDK that simplifies the development, and applications can scale both vertically and horizontally. The approach in [GRJ*09] is similar, but lacks of dynamic adaptation of the computation between mobile devices and the cloud. Cloudlets [SBCD09] and ISR [SKHH05] allow high abstraction and personalization of the computing environment by using VMs, but lack from fine-grained execution adaptation. [HuLe10] and [Mari09] approaches enable high horizontal scaling of the available ad-hoc mobile nodes, but with high communication overhead.

Table 3.1: Comparison of existing and proposed mobile cloud computing approaches

Application Model	Underlying Technologies	Cost Model	Programming Abstraction	Solution Generality	Implementation Complexity	Static Adaptation	Dynamic Adaptation	Network Load	Scalability
Offline	Vendor SDK	/	/	medium	low	high	low	high	low
Online	Web services, HTML5.0	/	high	low	low	high	high	medium	high
Chun and Mantatis [ChMa09] (CloneCloud)	Dalvik VM (Android)	in [ChMa10]	/	low	high	high	low	low	high
Satyanarayanan et al. [SBCD09] (Cloudlets)	VirtualBox Dynamic VM synthesis	/	high	low	low	/	/	low	low (vertical)
Giurgiu et al. [GRJ*09] (AlfredO and R-OSGi)	OSGi, Java	consumption graph	high	medium	low	high	low	low	medium (vertical)
Cuervo et al. [CBC*10] (MAUI)	.NET	linear optimization	high	high	low	high	high	low	high
Zhang et al. [ZJG10] (Weblets)	REST, C#	Naïve Bayes Classifier	high	high	low	high	high	/	high
Åhlund et al. [AMJ*09]	P2P	/	/	low	low	medium	/	/	/
Satyanarayanan et al. [SKHH05] (ISR)	VM, Distributed File System	/	medium	low	low	low	low	high	medium (vertical)
Huerta-Canepa and Lee [HuLe10]	Hadoop, XMPP	/	high	low	low	medium	low	high	high (horizontal)
Cao et al. [CJK*09] (Mobile WS)	Web services	/	high	medium	low	medium	low	medium	medium (vertical)
Marinelli [Mari09] (HyraX)	Hadoop	/	high	low	low	medium	medium	high	high (horizontal)

3.3 Mobile Multimedia

Mobile multimedia has become ubiquitous via a wide array of available content, services and devices. It profoundly changes people's habits and work practices as multimedia artifacts become integrated into their everyday lives. Shared multimedia experiences on different devices become common multimedia "prosumption" practices. Users (re-)create content into social situations to make meaning and value in ways not possible with traditional fixed computers and TV [OMVo07]. But the shift from fixed to mobile usage doesn't mean immediate identical user experience anywhere at anytime. Relevant research has been done in several domains related to mobile multimedia, and here only a few of them are surveyed which are important to this dissertation.

3.3.1 Metadata Collaboration

Technology could assist a group of people to communicate, manipulate and modify shared digital objects in a coherent manner [EGRe91], [Grun94]. Popular groupware software applications are based on simultaneous writing of a document by different authors, also known as shared file editing, instant chatting or cooperative design. Collaborative editing is the practice of participants within a group to work together to produce simultaneously a common output using a set of defined operations [LuMa09]. A lot of previous research deals with conflict resolution and avoidance in a concurrent collaborative editing session, i.e. when two or more users access the same document and perform simultaneous incompatible operations [LuMa09, Gerl07].

Since XML is the de-facto standard interchange and data format, systems that support real-time collaborative editing of XML documents could lead to a general solution for many application areas especially real-time collaboration. Furthermore, XML is widely used for

storing and exchanging multimedia metadata because its tree like hierarchical structure can describe the complexity of metadata descriptors which can be interchanged with different systems [BBD*08].

Currently, Operational Transformation (OT), first introduced by Ellis and Gibbs [ElGi89] then simplified by Jupiter System [NCD*95] is the most popular technique behind collaboration features. In OT, every client has a replicated copy of the document and sends the operations to the server side with an optimistic approach therefore the changes that are made on client side are reflected before sending them to the server. The main idea of the algorithm is to execute the locally generated operations without any delay, and then transform the remote editing operations into a new form according to the effects of previously executed concurrent operations while ensuring after every operation the consistency of the document at all participants [XSC*04]. Many research works improved the OT approach. dOPT [ElGi89], Jupiter [NCD*95], SOCT2 [SCFe97], GOT [SJZ*98] focus on linear structured text documents, whereas OT for SGML [DSL02] treeOpt [IgNo03] and P2P Editing on XML-like Trees [LuMa09] are some examples that performs OT on tree like structures. Furthermore, COT [SuSu06] and adOpted modified the algorithm by adding undo functionality. Google Operational Wave Transformation assures concurrency control of XML documents [WMLa10]. Consistency Maintenance Algorithm for XML (CMAX) [Gerl07] is a lightweight approach that focuses on ensuring collaborative editing on structural XML-based documents. The algorithm mainly inspired from operational transformation for structural documents. CMAX follows the same principles of OT for solving divergence and causal violations, however for solving the intention violation problem a new approach is applied. The OT update operation is not used since Gerlicher [Gerl07] showed that the probability of modification of the same tree node concurrently is relatively small when the XML tree is averaged size or big size. Therefore, CMAX delegates conflict resolution to the user instead of combining the attribute values by OT.

3.3.2 Communication for Collaborative Applications

The right set of underlying communication protocol is crucial in the mobile real-time collaboration. The XMPP protocol provides a pure XML foundation for real-time messaging, opening up tremendous possibilities for more advanced real-time applications. XMPP together with its extensions is a powerful protocol for collaborative services. Together they demonstrate several advantages beyond traditional HTTP-based Web services (e.g. SOAP and REST), such as decentralized, open and flexible (extensible) communication protocol, federation of servers, support for real-time data streaming in two directions, event notifications, remote procedure calls, and multimedia session management. Asynchronous invocation eliminates the need for ad-hoc solutions like polling.

Google Wave protocol is an excellent example of XMPP-based communication and collaboration platform for concurrently editable structured documents and real-time sharing between multiple participants. Novell Vibe Cloud [Nove11] is a web-based social collab-

oration platform for the enterprise providing social messaging and the ability of online document co-editing along with file management, groups, profiles, blogs and wikis, and security and management controls. Both Google Wave Protocol and Novel Vibe are very sophisticated collaborative editing software, but they rely on heavy-weight client JavaScript libraries which limits the usefulness for custom mobile applications.

On the other hand, the Collaborative Editing Framework for XML (CEFX) enables lightweight concurrent real-time editing of XML files using operational transformation algorithms [Gerl07]. Since the nature of XML is generic and extendable, different kind of information can be stored such as graphic files (SVG), AR contents (ARML), etc. Voigt [Voig09] further extended the framework by changing the communication protocol from JAVA RMI to XMPP. Moreover, communication data volume reduced significantly.

Similar to the Mobilis framework [SSSc10], our work also uses the CEFX+ for providing collaborative editing services and providing the communication between the nodes that attends a collaboration session on XMPP. However, the Mobilis Platform is heavy-weight at both client and server sides and also the platform lack of mobile AR browsing features. Junction [Stan11] is another XMPP-based communication framework for multi-device applications, but with limited collaborative features.

3.3.3 Mobile Multimedia User Experience

Despite the popularity of mobile video sharing, mobile user experience (UX) is not comparable with traditional TV or desktop video productions. Previous studies have reported on techniques for improving UX for mobile video. However, far too little attention has been paid to the practical realization under real-world application requirements. The issue of poor UX in mobile video sharing can be associated with the high development cost, since the creation and utilization of a multimedia processing and distribution infrastructure is a non-trivial task for small groups of developers.

Recently, video sharing has been extended on mobile platforms with considerable success. Since it is significantly cheaper and more convenient, mobile video has boomed in many domains like storytelling, live event streaming, practice sharing, video chatting, watching TV anywhere, etc. Many mobile applications for live video streaming from/to mobile phones reflect the growing interest in video-based sharing of life experiences in real-time.

However, we can observe three classes of mobile video discrepancies between users, content and devices. First, professional video content production shifts to higher resolution formats which are not suitable for common smartphones. For example, TV shows, sport games and movies are usually distributed as HD video (1,920 x 1,080 frame size in pixels), or soon, in the Super Hi-Vision standard (16 times sharper picture than HD TV). The resolution of mobile device increases, but their display capabilities will always be constrained due to the physical size of mobile devices. In effect, it means that more information is being captured than being displayed. Second, amateur video content shot with smartphones lacks

many characteristics that create the aesthetics of professional videos. For example, mobile video shots are often unsteady, without a smooth pan or zoom to the objects of interest and without clear shot segmentation. On the other hand, cinematographers carefully control the camera movement, intentionally control the lightning and edit the content in post production using their expertise. They achieve thus more appealing effects. Finally, video navigation in mobile applications follow the desktop metaphor. However, the limited screen size and mobile network bandwidth forbids fine navigation within a video using the timeline on the touch screen.

These discrepancies could lead to poor user experience (UX). The small screen size prevents users from recognizing video content details, especially in big-screen productions. The limited bandwidth causes problems in seeking or browsing the video, because it may take a long time to load the video.

Previous research on mobile video quality of service has focused mostly on network latencies or bitrate adaptation [SSWi07]. However, the perceived experience from the user and content perspective is neglected. Moreover, other works [STWD10, LiGl06] deal with content-based video retargeting algorithms for improving mobile video UX, but applied only to locally stored videos. One major limitation is that they neglect the complexity in development of mobile video sharing applications.

The development of mobile video sharing applications and services, however, comes at high cost. For instance, application developers need constantly to deal with the format and resolution “gap” between Internet videos and mobile devices. Another example of the issue is the provision of up/down streaming infrastructure. Most of these automatic approaches and techniques require substantial processing power, highly-specialized software tools and components for indexing and adaptation of video content. We are at the early stage of confluence of cloud computing, mobile multimedia and the Web. Developers should be relieved from the cumbersome infrastructure setup in the multimedia chain in mobile-Web settings. They should be able to focus on improving the mobile UX within their apps.

As mentioned before, user experience is a very important part of mobile video viewing. Some of the problems previously discussed have been tried to overcome and to be analyzed recently. With the issue of small screen size, some propose to use zooming, ROI enhancement and bit rate increase as a solution. A recent study shows that the percentage of persons wanting to watch sport events on the TV is equal with the percentage of persons wanting to watch it on mobile devices. But the problem with mobile devices is that they offer a small viewing size and bit rate limitation which results in an unpleasant viewing experience. Song et al. [STWD10] did a study on the impact of ROI enhancement, zooming in and bit rate increase on the mobile user experience with 40 participants. Most of them complained about the poor quality of sports streams on mobile devices and that it is very hard to follow the ball on the screen. Song et. al. [STWD10] present a system which is able to do ROI enhancement, zooming in and bit rate increase. For zooming in it was found that a factor between 1.13 and 1.44 is the preferred one. They chose a soccer match and a talk show to test four approaches which include the three named ones plus the original video. Zooming is

done from the center. Their result was that the increase of bit rate does not have any impact on the mobile user experience. For videos with human faces and slow motion ROI and zooming have both an equally positive impact on the overall experience. In comparison to it, the experience for fast videos is only improved by zooming. Zooming is a good solution to overcome the problem of small screen sizes. In addition, Knoche et al. [KPSV07] present a function to compute the optimal zoom depending on the device's screen size for extreme long shots. This solution additionally improves the mobile user experience.

As mobile devices are often transported to different locations the bandwidth is changing all the time. Therefore the mobile multimedia services should adapt to the current environment. Other parameters are actual screen size, remaining battery power etc. Papakos et al. [PCRo10] propose VOLARE — a context-aware middleware-based solution — that monitors the resources and dynamically adapts cloud service requests accordingly. The main aim is resource-efficient and reliable cloud service discovery. Their prototype implemented a series of adaptation policies. The dynamic adaptation of service requests yields an improvement on service discovery of cloud services and binding resource efficiency and cost-effectiveness which is very lucrative for mobile providers, too.

Another issue that mobile users have to deal with is browsing videos. Bursuc et al. [BZPr10] proposed a interesting approach as an mobile video browsing and retrieval problem. The system allows to browse and access video content in finer, per-segment basis. Segment information is saved using MPEG-7 descriptors. As videos often have a huge amount of heterogeneous information it is important that the user will be able to easily access segments of interest. The video segmentation itself is done on three levels including scenes, shots and key frames.

A possible solution for browsing and personalization of video streams is by using annotations which can be generated automatically or manually. Patrikakis et al. [PPS*11] use the concept of semantic annotation and metadata to identify events, favorite players and other concepts. Their main goal was the personalization of multiple channel video dependent of the user preferences. In a scenario where a user is watching a football match using his mobile phone but has to choose from several streams showing different camera angles and, therefore, various events, annotations are used to improve the video browsing experience by showing the metadata as tags and allowing navigation by them. Other different approaches to improve the user experience of mobile video using semantic annotation are also existing. The TuVista [BeGr09] project concentrates on the social and emotional aspects of user experience. TuVista is a system which provides nearly live sports content viewing. It provides the possibility of multiple live videos and push notifications with XMPP. One of the main aims of this system was to reduce the content publication time from 15 minutes to 30 seconds. Therefore, the editing tool is the heart of the system. In the first phase of the prototype two persons had to edit the video but in the second phase a new editing tool was introduced which only requires one person to edit the video. During the on-site user studies at a stadium and at a volleyball game was found out that mobile sport clips make sense only with maximum delay of 30 seconds just after the real event happened. For example,

Mobicast [KERE09] is a system for mobile live video streaming that enables collaboration between multiple users streaming from the same event using their mobile phones. They are stitching the different video streams into one panoramic view to improve the viewing experience. The tasks of video casting, content analysis, casting director and provisioning and streaming is all done in the cloud.

Most research concentrate either on the encoding/transcoding site or the resources. Improving user experience should not only be done by encoding. It also has to be adapted to the current environment. User experience changes considerably with screen size and other resources like battery life.

3.4 Experiences from Building Mobile Multimedia Community Services

The design of core parts of CAELUS architecture has emerged as result of literature research and extensive experiences with Web-based systems and mobile applications within our research group. This section gives a short overview of several research prototypes which preceded the work covered under this dissertation. Within these projects, developed within our research group, I have participated at development, operational or evaluation levels. The lessons learnt and experiences gained within these projects served as an input to defining important groups of concerns and requirements regarding the design, implementation and operation of mobile multimedia community services.

LAS: Lightweight Application Server

On the backend side, our research prototypes are driven by the Lightweight Application Server (LAS) [SKJR06]. LAS is an extensible lightweight middleware server that follows the service oriented architecture (SOA) principles and uses HTTP and SOAP based client-server communication. It is a platform-independent Java implementation that can be flexibly (re-)combined among various tools and communities. The server functionality can be extended based on community's specific needs. In addition, LAS components provide functionalities such as the connectors to various databases. LAS is equipped with core services for user and community management, access control, security and authorization mechanisms. Service developers can extend the functionality of this community engine by creating and mashing up new services. Consequently, LAS has been used rapid prototyping in many use cases of mobile communities such Virtual Campfire and SeViAnno, described in the forthcoming paragraphs.

Virtual Campfire

Virtual Campfire (VC)[CKJa10] embraces a set of advanced applications for communities of practice. It is a framework for mobile multimedia management concerned with mobile multimedia semantics, with multimedia metadata, with multimedia context management, with ontology models, and with multimedia uncertainty management. The LAS community engine facilitate simple user and community management. Multimedia services are easily used for multimedia semantics and context management, including metadata management. A common data repository is shared for cultural heritage management as well as technology enhanced learning. These domains have high requirements on mobile multimedia management. The common data repository lowers the barriers to develop various cultural heritage related applications. For instance, a context-aware adaptation service is realized to facilitate the adaptation process of multimedia artifacts based on mobile community context [CKHJ08]. Context information includes geospatial, temporal, community and device context. On the one hand, context modeling is applied to represent context information, while context reasoning is applied to enhance multimedia semantics. Mobile device context information is used for multimedia adaptation in order to enhance mobile access of appropriate multimedia. Context information is applied to reduce uncertainty aspects and to enhance multimedia query results. Moreover, this service focuses on the presentation and management of context uncertainty.

SeViAnno: A Semantic-enhanced Video Annotation Service

SeViAnno [CRJ*10] is an MPEG-7 based interactive semantic video annotation Web platform with the main objective to find a well-balanced trade-off between a simple user interface and video semantization complexity. It allows standard-based video annotation with multi-granular community-aware tagging functionalities. Various annotation approaches are integrated and depicted in Figure 3.9. The main elements of semantic information in videos are persons, places, time, buildings and objects. LAS provides a range of operations on MPEG-7 descriptions including their persistence in a native XML database. Several multimedia services are employed. The MPEG-7 Semantic Base Type Service is designed for the management and persistence of MPEG-7 semantic base types, which can act as semantic tags assigned to multimedia or segments. The MPEG-7 Multimedia Content Service is designed for annotations of complete multimedia files such as images and videos. Those services using the segmentation of videos employing the Audio Visual Segment Temporal Decomposition Type of the MPEG-7 standard have been added to the existing multimedia services. The Audio Visual Segment Type is used to add the semantic references of the base types corresponding to the video segments. To save the time point and the duration of the segment the MPEG-7 Media Time type as a description tool has been employed.

SeViAnno All Videos Upload video Logout

Gandhara Film



Description (edit)
Gandhara - Land des Friedens. Zur Ausstellung in Bonn.

Tags (edit)
Gandhara, Gandhara, Buddha, Bamiyan, SBG Aachen, Taliban, Peshawar, Frieden, Relief, Siddharta, Toubekis, Afghanistan, rwth,

Add semantic annotation
Agent Concept Event Object Place Time

Semantic annotations (43)

Type	Annotation	Time	Creator
🕒	datum 2009-09-22	01:09	admin
📍	India	01:17	admin
🌐	Iran (31.233°, 58.961°)	01:19	admin
🌐	new dehli (5.3679°, 115.46°)	01:19	admin
✈️	Map Flight	01:22	mlottko
🌐	Siddharta Gautama (°, °)	01:40	admin
🌐	Peshawar (34.004°, 71.545°)	01:45	admin
📍	Map	01:48	mlottko
🌐	Gandhara (33.992°, 71.480°)	01:56	yiwei
🕒	5. Jahrhundert vor Christus T00:00:00	02:05	mlottko

Place annotations



Figure 3.9: SeViAnno - A Web application for multimedia semantic annotation

Lessons Learnt

By providing core services such as user and community management, security and authorization, service developers (students and researchers) were able to rapidly prototype new advanced services and provide application solutions in different domains. Moreover, by following the service-oriented approach, we were able to satisfy different stakeholders and requirements. For example, same semantic multimedia services have been successfully used for applications in cultural heritage and technology-enhanced learning. Therefore, CAELUS embraces the same approach of providing core services and a service-oriented architecture, but with the difference of being designed for and deployed on a cloud infrastructure.

Mobile multimedia applications should be able to handle large data sets and many users. The reason is that the rise of mobile multimedia, social media, and Internet of Things increases the volume and detail of information that applications need to handle. With the given server-based architectural solution, we noticed that changes are needed in order to be able to create mobile real-time collaborative applications with a capacity to scale on large datasets and many users. LAS services performed well on small set of user clients, but performance degraded when a server instance needed to serve myriad requests to the multitude deployed services. Similarly, standard Web applications fail to meet these requirements as well. Vertical scaling with more hardware solves the issues to a certain amount, however, a more elegant way is to enable the services with scalability and real-time collaboration primitives from the beginning.

The SeViAnno use case revealed that only a Web interface for multimedia services is insufficient. Professionals from the cultural heritage domain expressed needs to operate with multimedia artifacts during field trips. It was clear that the services need both mobile and Web endpoints for users which can work in a synchronous way. Novel protocols and standards like XMPP, SIP, or HTTP DASH tend to overcome some of the issues in the dynamic mobile/Web ecosystem. The HTML5 standardization efforts consider the requirements for interactive applications that can run both on traditional computers and smartphones, thus bridge over the gap between them. Streaming media and metadata, HTML5, XMPP, and WebSockets have great potential to empower users' rich multimedia sharing experiences across Web and mobile devices.

In summary, intrigued by the success of the cloud paradigm in enterprise settings, we have considered a cloud-based multimedia framework for mobile and Web services using emerging protocols and standards, partially described in [KCK10, KRKC10]. The goal is to enable a single person to design and run Web-scale multimedia applications with little effort. In addition, to enable operability in outdoor environments with limited or no Internet infrastructure, we needed to seek for mobile app solutions that able to operate smoothly under volatile network connectivity.

3.5 Summary

This chapter gave an overview of relevant research in three related fields of mobile multimedia services. In particular, the surveyed works fell into the intersection of these three fields as these works actually drive the innovation further in cloud computing. On the basic level, the development of multimedia cloud services needs an infrastructure design based on cloud principles (see Chapter 2), and at the same time, specialized for multimedia formats, delivery mechanisms and processing algorithms. Moreover, the multimedia formats and protocols should embrace the distributed nature of cloud infrastructure and resource impoverishment of mobile devices and networks. To gain additional value from the cloud computing paradigm, we need to explore alternative computing models beyond the traditional client/server model. These new models incorporate several factors within the system architecture, such as mobility, resource-constraints and cloud scalability etc., and eventually resulting in network-resilient, offline-proof, adaptive mobile applications. Finally, the chapter indicated that mobile multimedia gains momentum in research and commercial products but several aspects need to be considered such as collaboration and user experience.

The papers and projects elaborated here were surveyed since they overlap to certain extent with the mobile multimedia services and prototypes of this dissertation. Many of them excel in providing certain features that focus only on a limited scope. Within CAELUS, we tried to achieve comparable quality, but additionally to provide a comprehensive platform for a variety of mobile multimedia services needed for the development of next-generation

mobile applications (see Chapter 5 and 6). Many ideas and guidelines for CAELUS derive from our experiences with advanced mobile community information systems and lessons learnt, as described in Section 3.4. In the next chapter, a comprehensive conceptual model is given, which is based on requirements drawn from the state-of-the-art research works and real-world scenarios.

In the province of the mind, what one believes to be true either is true or becomes true.

John C. Lilly (1915 - 2001)

Chapter 4

Mobile Multimedia Cloud Computing

This chapter extends the theoretical foundations presented in Chapter 2 towards universally-applicable software architecture models of complex mobile services as a prerequisite for building multimedia cloud applications. The chapter attempts to guide the development process on how to deploy and apply the cloud computing paradigm in applications to achieve the intended scalability, performance, interoperability, etc. In general, this development process requires understanding of cloud concerns and mobile multimedia issues. To systematically analyze the their complex interplay, the areas of concern and issues are grouped under three facets, i.e. technology/system, mobile multimedia and user/community. This three-faceted view is elaborated in the first part of this chapter where the facets are further segregated into sub-perspectives. Successively, the second part of the chapter derives three reference models that are essential to mobile multimedia application architectures. The models are complementary to each other and are optimized to specific classes of services. The chapter finally ends with a feature comparison between the three mobile cloud computing models. The applicability of these ideas is shown within several domains, which is presented in the subsequent chapters (Chapters 5 and 6).

4.1 Faceted View of Mobile Multimedia Clouds

Cloud-based applications exhibit properties of complex adaptive information systems. Various stakeholders interplay in the creation process of mobile multimedia services. Moreover, many emerging and advanced technologies are available to enrich mobile multimedia experiences. Loosely inspired by the idea of the three-faced view of information systems [DDJ*98], we, too, propose three crucial facets of the outlook of mobile multimedia clouds: technology/system, mobile multimedia, and user/community facets. Each facet represents a broad view on ways to effectively satisfy the requirements from different perspectives. The three facets reflect complimentary perspectives for the analysis and realization of mobile multimedia cloud services. This faceted view is useful since it helps to conceive cohesive and complete solutions. The three facets are derived from practical experiences and litera-

ture research. It was clear from the literature surveys described in previous chapter that the research communities tend to focus only on one or two of the three facets.

Each facet has further sub-perspectives related to the design decisions and realization. Perhaps, they are not complete analysis of mobile multimedia cloud computing; however, they give a sufficient and complimentary coverage with useful guidelines for its realization. The system facet is concerned with technologies and approaches used to build a system that performs given tasks. The mobile multimedia accounts for multimedia specifics when applied to mobile settings. Finally, the user/community pertains to work practices, policies, organizational and usability issues to accomplish certain task. We have chosen these facets because technology drives innovation in new services, multimedia is a central artifact in today's digital world, and users and communities are the main stakeholder since the Web 2.0 digital revolution.

4.1.1 System and Technology Facet

The system and technology facet founds a basic ground to facilitate mobile cloud computing for application portability and platform independence. Internet & American Life Project and Elon University's "Imagining the Internet" Center have conducted a survey showing that over 71% of the subjects think that most people will work in Internet-based cloud applications such as Google Docs and in applications run from smartphones by 2020 [AnRa10]. Mobile devices will be the driving force for people to make use of cloud-based services and applications. However, there are still technological barriers to use cloud services on capacity-limited mobile devices.

Data Management

With the growing scale of Web applications and popularity with mobile applications three trends can be observed in terms of data management requirements. First, large data becomes associated with many applications limited not to scientific domains only, but ordinary user applications such as photo sharing. Therefore, scalable cloud data management becomes a necessary part of the cloud ecosystem. Some of the popular scalable storage technologies in the moment are Amazon Simple Storage Service (S3) [AmaS3], Google BigTable [CDG*08], Hadoop HBase and HDFS [Hado09], etc. Basically, these distributed blob and key-value storage systems are very suitable for multimedia content, i.e. they are scalable and reliable as they use distributed and replicated storage over many virtual servers or network drives.

Second, applications begin to differentiate between themselves in terms of their distributed storage priorities. Traditional distributed data storage systems are based on the consistency premise, but some novel applications are willing to sacrifice consistency for availability. For example, an online shop application would prefer to process as many as possible orders even if the backend system would require more time to achieve consistency. Applications

need to understand their priorities, since it is impossible to achieve *Consistency*, *Availability* and *Partition-tolerance* (the CAP theorem) at the same time [GiLy02]. Meanwhile, recent research works report on data management tools with elastic (scale up and down) consistency which can be based on different parameters (application requirements, load, and cost) [AFP*09, BBC*11]. Regarding multimedia metadata management, the aforementioned techniques are still not enough explored compared with traditional relational databases and ontologies. As these data storage technologies fall into the category of NoSQL databases [Catt10], they trade off the schema, joins, and ACID transactions for elastic horizontal scaling and big data storage. The ACID transaction principle refers to execution of transactions that must be guaranteed with atomicity, consistency, isolation, and durability. This is rather more challenging for the distributed data storage in the cloud than centralized database management systems (DBMS).

Finally, beyond the scalable and highly available cloud storage, pervasive and ubiquitous applications require content to be shared between users, devices and services in a timely fashion. In addition, mobile data management requires support for a disconnected operation, since it needs to cope with intermittent and limited network connectivity, tight power constraints and a number of fault tolerance issues. As a result, service development requires a large effort for application's data storage and synchronization layers. Common solutions are based on asynchronous interactions (to hide the cost of remote operations), client-side read caching (to reduce bandwidth and latency, while bounding data staleness), batching of write operations (to reduce power costs), protocols to arbitrate concurrent and disconnected operations on shared data, timeouts and reconnection strategies (to handle network coverage issues), notification mechanisms (to allow data to be pushed from the server side), and manual sharding of data and replication (to guarantee server side scalability and fault tolerance) [CCS*12]. However, to facilitate rapid prototyping of mobile cloud services, a programming abstraction of such operations is needed which should basically support logical data providers that span between the devices and cloud, and at the same time, perform dynamic client-side caching, notification mechanisms, reduction of exchanged messages and energy-awareness.

Communication

The communication sub-perspective imposes requirements for anywhere anytime access to data and information, support for real-time data transport (such as streaming or messaging), high-level application protocols and ability to adapt network configurations dynamically (e.g. interconnect virtualized computer clusters in data centers). Mobile multimedia clouds require broadband Internet connections with high data rates, with low latency and rely on always-on connectivity in order to meet the required quality of experience (QoE). Novel network infrastructure such as 4G mobile networks with increased upload speed of 500 Mbit/s and download speed of 1 Gbit/s [PaAs09] open new classes of interactive applications such as instant streaming and sharing of high-quality videos [GKFu10], conferencing and

remote-rendered 3D content [PHE*11]. The low latency would allow users to benefit real-time interaction in games and other resource-hungry applications.

In any basic mobile multimedia cloud scenario, the platform needs to provide basic services such real-time sync, push, messaging, etc. The sync service synchronizes all state changes between the devices and the cloud provider. For example, this could include a shared application state, user preferences/settings or content modifications. The push services propagate changes and delivers messages to different end points in a timely and energy-efficient manner. For example, any network application would benefit from a lightweight mechanism that can notify the devices about certain events such as process state change (start, finish, pause, etc.), and at the same time, avoid persistent checking on the other communication side. These push notification mechanisms affect the energy usage on the battery-powered devices. Since both mobile and desktop applications gradually become an amalgam of complex and interconnected services, a scalable and extensible messaging infrastructure becomes a necessary requirement in any cloud platform.

To facilitate such services, cloud platforms need to embrace post-HTTP application protocols such as XMPP [Sain11], SIP [SIP02], or WebRTC [WebR12]. These communication protocols together with their extensions are powerful solutions for cloud services that demonstrate several advantages beyond traditional HTTP-based Web services, e.g. SOAP and REST [KCK110]. For example, XMPP provides a common layer to connect human-to-human, human-to-machines and machine-to-machine synchronous and asynchronous communications [HoWa10]. Moreover, scalability (in terms of throughput required per number of nodes) and traffic overhead of XMPP outperforms SOAP/HTTP, as shown in [AlMa10].

Computation

Looking from a computation sub-perspective, mobile multimedia clouds need to provide to mobile devices a transparent access to the “unlimited” resource pool residing in cloud data centers. Moreover, multimedia computation tasks should be encapsulated (most commonly in virtualized containers) to be executable on a variety of both mobile and static hardware. Clouds have a huge processing power at their disposal, but it is still challenging to make it truly accessible to mobile devices. The traditional client-server model and Web services/applications can be considered as the most widespread cloud application architecture. However, several other approaches to augmenting the computation capabilities of constrained mobile devices have been proposed.

Integrated mobile cloud solutions to surmount mobile devices’ shortcomings by augmenting their capabilities with external cloud resources are needed. The full potential of mobile cloud applications can only be unleashed, if computation and storage are offloaded into the cloud, but without hurting user interactivity, introducing latency or limiting application possibilities [KCK110]. These solutions need to enable mobile application developers to

have the illusion of programming on much more powerful mobile devices with higher computational and storage capacities.

The mobile devices landscape features heterogeneous hardware and operating system run-times. This imposes challenges to enable seamless computation interoperability. Several approaches are feasible to surmount such issues. Representative examples include the execution of software images on virtual machines into the cloud [ChMa09] or into nearby computers [SBCD09]. Instead of offloading the whole mobile software stack, some propose offloading of application parts as computation tasks [Kris10]. The applications could be automatically split [KPKB10], or developed intentionally for an adaptive shift of its execution between a cloud and a device [ZKJG11]. Recent studies have shown that offloading can efficiently save energy [KuLu10, CBC*10] and increase performance [OYZh07] by an order of magnitude on common mobile platforms.

4.1.2 Mobile Multimedia Facet

The mobile multimedia facet pertains to aspects related to multimedia itself as a rich resource for multimedia processing. It is related to how multimedia is presented or encoded at a lower level, and analyzed, modeled or processed at a higher level. This facet is addressed along with the high growth of mobile multimedia traffic. For instance, mobile video traffic was 52% of all mobile traffic by the end of 2011 and this growth will continue to become over 80% in 2016 [Cisc12]. Therefore, optimizations in the complete multimedia life cycle adhering to cloud infrastructures are needed.

Content Adaptation

Multimedia applications require transferring adapted multimedia through different interconnected networks, servers, and clients with different media modality and quality. The transformation of content to a suitable presentation occurs typically at two phases, i.e. at the selection of media version based on user preferences and at the adjustment according to the computing environment context. Multimedia content is usually compressed using compression algorithms or codecs, in order to achieve smaller file size for faster transmission or more efficient storage. However, different mobile device media platforms are based on different formats, containers and coding. For example, if we consider video codecs, Android OS supports H.263, H.264 AVC, MPEG-4 SP and VP8, while iPhone iOS supports H.264 and MPEG-4. Obviously, in order to achieve interoperability in the heterogeneous ecosystem of mobile platforms, adaptation services are needed. In general, video adaptation requires large computing resources, especially, under heavy request rates. Clouds tend to abstract the technological complexities connected with seamless multimedia content adaptation. For instance, cloud software-as-a-service (SaaS) solutions such as zencoder.com [Zenc13] and encoding.com [Enco13] have emerged, which can do the heavy lifting of CPU-expensive video encoding, thus relieving clients from any upfront investment.

Transcoding and transrating are only one option to adapt multimedia content to be consumed on mobile devices effectively. Other possibilities exist in modification of the content without losing important information, known as content retargeting techniques. For example, to detect regions of interest in a video stream and deliver a version of the same stream only cropped to the ROI. Another example is the process to detect relevant objects (and timepoints) in a video and then provide navigation cues to the objects which should result in decreased video browsing workload. All these kinds of services require a complex setup of computer vision and machine learning tools. Moreover, service developers need to access service interfaces from both devices and Web applications.

Multimedia Modeling

Many advanced multimedia services such as adaptation, personalization and filtering require some kind of modeling of semantic and structural information about the content. Multimedia content is not self-descriptive by itself, therefore, metadata plays a central role in mobile multimedia applications. In general, metadata describes different aspects of the context, computing environment, user preferences, domain knowledge and relationships between media artifacts and content semantics, in order to provide value for the end user. Therefore, strong modeling foundations to express these complex relationships are needed.

At first, content interpretation is required. Models typically include objects detected within the media and their properties, spatial relationships between these objects, events involving objects, and temporal relationships between objects and events [Ange06]. These model elements are mapped into some kind of structure. For instance, mobile devices are able to produce different kinds of multimedia content. Moreover, those rich sensing functionalities embedded in mobile devices provide valuable context information which can be used for indexing, querying, retrieval, and exploration of the multimedia content. Multimedia metadata standards such as MPEG-7 and RDF ontologies are the foundations for semantic multimedia knowledge representation and interpretation [CKKo09]. Multimedia models can also benefit from the rich sensor data coming from mobile devices and integrate it into the models.

Multimedia Semantics

Access and management of multimedia content and context relies intensively on semantic descriptors. Low-level multimedia analysis such as feature extraction, metrics and segmentation can be automatically processed. Multimedia artifacts can be represented with shape-based or texture-based features and used for visual content description or classification in general categories. The output values of these attributes help mobile clients to process multimedia files of various formats easily.

Models based on standards are usually capable of including only measurable quantities, which might not always suffice for context-aware and intelligent multimedia applications.

Recent research is focusing on more complex models which are useful for reasoning and fuzzy logic systems. Multimedia analysis, machine learning methods and logic-based modeling have proved to be good for discovering complex relations and interdependencies, which are serving as input for reasoning in the media interpretation processes.

However, the approaches depend on the availability of large training corpus of labeled multimedia data and metadata, which are difficult and expensive to produce. On the other hand, there has been an explosion in popularity of social tagging and annotations of multimedia artifacts as part of the Web 2.0 phenomenon. They are community-based approaches to classifying multimedia on the Web, which also enhance the discovery and re-use of multimedia across communities and disciplines. The result is more relevant, light-weight and cheaper metadata than traditional cataloging systems. Nevertheless, there is a need of holistic approaches that take the advantage of each individual approach and combines them to produce the best result for the end user.

4.1.3 User and Community Facet

Over the past decades, new media, new technologies and devices, new ways of communication continuously define new formats of practice of professionals or knowledge workers. For instance, the world wide access to heterogeneous information over the Internet has created new means for cooperative work. Social software well known by examples like the digital image sharing platform flickr.com [Flic06], the digital video sharing platform youtube.com [YouT07] or the social bookmarking platform delicious.com [Deli13b] can be broadly defined as an environment that support activities in digital social networks [KIJ08b]. Professionals change their work styles according to the new possibilities.

This facet is related to user and community experiences with regard to mobile multimedia applications. Pew Research Center reported that 71% of online adults used video-sharing sites such as YouTube and Vimeo as of May 2011¹. On those video-sharing sites, users and communities are the main actors who produce and consume multimedia on mobile devices. The previous two facets are addressed to mobile multimedia service developers, while this facet is more related to end-user requirements. User interfaces are the access bridge between users and applications. In addition, user evaluation procedures can be well applied to test this facet.

Sharing and Collaboration

People as social beings by nature usually like to interact with each other. Meanwhile, the capabilities of mobile networks and devices craft new ways of ubiquitous interaction over Web 2.0 digital social networks. Consequently, mobile devices, Web 2.0, and social software result in two phenomena. First, there is an exponential growth of user-generated

¹<http://pewresearch.org/pubs/2070/online-video-sharing-sites-you-tube-vimeo>

mobile multimedia on Web 2.0 which, as a result, is a driving force for further mobile device improvements. Second, there is a large number of diverse emergent communities, i.e. groups of people, usually co-workers or groups of people who have similar interests trying to perform some tasks to achieve a common goal.

These two phenomena clearly show the demand for easy creation, sharing, and collaboration of multimedia elements such as photos, videos, interactive maps, learning objects, etc. Fortunately, sharing is an indigenous part of cloud services. Cloud computing acts as an enabler and accelerator of collaboration services. Sharing through a cloud generally enhances the quality of service, because cloud-to-device connections tend to be better than device-to-device connections.

Ubiquitous Multimedia Services

Ubiquitous and pervasive computing is the post-desktop model of human-computer interaction where the information computing and processing functionalities are interwoven within everyday activities and objects. Any new mobile application has to anticipate the consumption and production of multimedia in a variety of contexts. Recent advancements in mobile wireless technology has spurred research interest in this area. Tools and methodologies for design-oriented approaches that can capture the user experience with this technology are needed.

One of the biggest challenges in future multimedia application development is device and network heterogeneity. Future users are likely to own many types of devices and connect online from any location at any occasion. One-quarter of mobile users are predicted to own two or more mobile-connected devices by 2016 [Cisc12]. Switching from one device to another users would expect to have ubiquitous access to their multimedia content. Seamless roaming of multimedia session between device to device, network to network and user to user is needed. Moreover, mobile multimedia services need to be accompanied with mobility and location management. Cloud computing is one of the promising solutions to offloading the tedious multimedia processing on mobile devices and to make the storage and delivery transparent.

Privacy and Security

The adoption of cloud computing has unique security and privacy implications in mobile information systems. The aspects are related to ensuring that the data and processing controlled by a third party is secure and remains private, and the transmission of data between the cloud and the mobile device is secured [Lage11]. Clouds provide access to data, but the challenge is how to ensure that only authorized entities can access the data. This requires a combination of technical and non-technical means, i.e. clients need to trust their providers and the providers need to ensure their technical competence and integrity (e.g. through certification and service-level agreements).

Holistic trust models of the devices, applications, communication channels and cloud service providers are required [Pear09]. The responsibility for privacy and security is shared between the providers and consumers. Sharing of responsibility is different at different service models. For instance, at IaaS level, the provider handles only some low level data protection capabilities whereas the consumer is responsible to secure the running operation system, execution environment and content. In contrast, at SaaS level, the provider carries out the bulk of security responsibility.

4.1.4 Summary of Facets

In summary, all these three facets make a good structure for how to take various complex aspects into consideration at developing mobile multimedia cloud architecture. Although they have some overlapping sub-perspective as well, the foci are distinctive and a combination of different sub-perspectives can refine the requirements. For example, the ubiquitous multimedia services facet is tightly related to the mobile multimedia facet as well. From the technology facet perspective real-time communication protocols are related to provisioning of servers; whereas from the user and community facet perspective they are related to realization of real-time collaboration. Furthermore, MPEG-7 and ontologies enable expressing multimedia semantics. The combination of MPEG-7 and real-time protocols enables easy creation of scalable semantic multimedia real-time collaborative applications. Chapter 5 and Chapter 6 exemplify the three facets. Table 4.1 summarizes the afore-discussed facets and their sub-perspectives.

4.2 Application Reference Models

The understanding of the next-generation multimedia services starts with the investigation and examination of mobile and cloud architectures. The following sections provide details about reference models that incorporate cloud principles discussed in Chapter 2, and also cover the requirements identified earlier in this chapter. Each reference model from this list emphasizes certain properties of cloud computing and exhibits functionalities specific for a certain operating environment (i.e. data center, mobile device, network, etc.) Clustering into three main reference models enables us to understand the benefits and drawbacks of each of them.

Moreover, certain modeling tools can be used to visualize the relationships between different actors, stakeholders and the respective socio-technological system requirements. Here, I used the *i** modeling framework [Yu97b] which is able to capture the intentions and strategic dependencies. In *i** actors act intentionally by having goals, competencies, commitments, needs, desires, and know-how and resources for achieving goals. Agents act strategically, i.e. agents depend on other agents and need to cooperate between each other in the process of achieving their goals.

Table 4.1: Summary of the three facets and their sub-perspectives

	Sub-perspective	Challenge	Opportunity
System/technology	Data management	Large-scale data	Analytics, distributed FS
		Content availability vs. metadata consistency	NoSQL, distributed DBs vs. traditional RDBMS and XML DBs
		Synchronized data	Abstraction of logical data providers using caching, notification mechanisms and energy-awareness
	Communication	Anywhere anytime access	4G and converged fixed/mobile wireless networks
		Real-time	Post-HTTP protocols (SIP, XMPP), push mechanisms
		Dynamic network configurations	Software defined networking
	Computation	Computation migration	Encapsulation (e.g. via virtualized containers)
		Elastic workload	Scalable architecture
Mobile multimedia	Content adaptation	Compression codec interoperability	Transcoding
		Network bandwidth limitations	Transrating
		Rendering limitations	Intelligent adaptation based on ML and CV techniques
	Modeling	Semantic and structural information expression	Metadata and ontologies
		Content interpretation	Object, event detection, structures for spatio-temporal relationships
		Context expression	Harvesting of rich sensor data
	Semantics	Low-level semantics	Feature extraction, metrics and segmentation
		Labeled multimedia training data	Web 2.0 social tagging, collaboration
		Multimedia interpretation	Machine learning, OWL reasoning, fuzzy logic
User/community	Sharing and collaboration	User-generated data	Web 2.0 sharing principles
		Multimedia collaboration	Real-time services, community awareness
	Ubiquitous multimedia	Spatio-temporally-scattered computing	Session mobility and seamless roaming
		Multi-device computing	Distributed user interfaces
	Privacy and security	Data confidentiality	Access control lists, certifications, service level agreements
		System integrity	Holistic trust models

Such conceptual models can capture the social and intentional dimensions in multimedia cloud service development which are comprised from many actors: *Cloud Providers, Service Providers, Content Providers, Network Operators, End users and Communities of Practice*, etc. These actors have been explained and exemplified throughout this dissertation on several occasions. Agents can be used throughout the conceptualization, requirements analysis, design and realization of complex socio-technological systems. In agent-oriented modeling, systems and elements are only partially knowledgeable and controllable. i* modeling framework was chosen because it can characterize the relationships between agents at intentional level. In fact, new information systems exhibit technical (hardware and software interactions) and social complexities (loosely-coupled network of actors). An agent has its own initiative, and can act independently. Consequently, for a modeler and from the viewpoint of other agents: its behavior is not fully predictable, it is not fully knowable, nor fully controllable [Yu97a]. For example, cloud providers aim to provide cost-effective general-purpose computing and storage platforms which can be accessed through the Internet so they can charge for the used resources in a utility-based manner. However, the cloud data center details are hidden from the other actors (e.g. service providers) who cannot always predict the performance of the cloud infrastructure nor control it. Nevertheless, cloud providers and service providers can relate to each other at an intentional level via multi-lateral relationships. Such relationships form an unbounded network where cooperation plays a major role but always striving to meet own goals.

The i^* framework supports two types of models, i.e. strategic dependencies (SD) and strategic rationale (SR) models. SD models are used for modeling intentional, strategic relationships among actors in a form of an actor diagram. SR models capture the individual rationale behind dependencies, and analyzes alternatives and dependencies fulfillment by a goal diagram. The main syntax elements of i^* are: *Actors*, *Actor Associations*, *Goals*, *Softgoals*, *Tasks*, *Resources* and *Links*. Actors linked between each other indicate that one actor depends on the other for something in order that the former may attain some goal. The depending actor is able to achieve goals by realizing the object around which the dependency centers. There are several types of dependencies: *goal dependency* (an actor depends on another actor to make a condition in the world to become true), *task dependency* (an actor depend on another to perform an activity), *resource dependency* (an actor depends on another for the availability of an entity), and *softgoal dependency* (a variant of the goal dependency but where there are no hard criteria for what it constitutes a goal). The SR model provides more detailed view of the actor's inside relationships. Intentional elements (goals, tasks, resources, and softgoals) appear also as internal dependencies arranged in hierarchical structures of means-ends, task decompositions and contribution relationships.

4.2.1 “Cloudified” Server

The definition of cloud computing from Chapter 2 highlights that clouds are a managed pool of servers in data centers. Enabled by virtualization technology clouds strive to provide infrastructure as a service which can be used by arbitrary cloud customers with minimal efforts. At the outset of cloud idea, cloud services were used mostly for Web 2.0 business applications (such as e-commerce sites) or Web-scale document processing (such as Web search engines). The next step was to deploy multimedia applications in the cloud. But media operations are more complex. The easiest way is migrate the server to a cloud-hosted virtual machine. Such architectural model benefits from application consolidation, meaning that all application components (database, Web server, etc.) can be hosted on one physical machine. It benefits from the ability to migrate between different hardware machines and recover from fail-overs. However, such an architecture lacks other core cloud features: scalability and elasticity.

To enable multimedia applications to scale up and down with the demand, the software architecture needs to be built around the distributed and parallelized cloud infrastructure. Throughout the dissertation, I name this architectural model as “*cloudified*” *server model*. It denotes a model where the services are delivered to mobile devices using the traditional server-client model, but cloud computing concepts are applied to the “server” architecture.

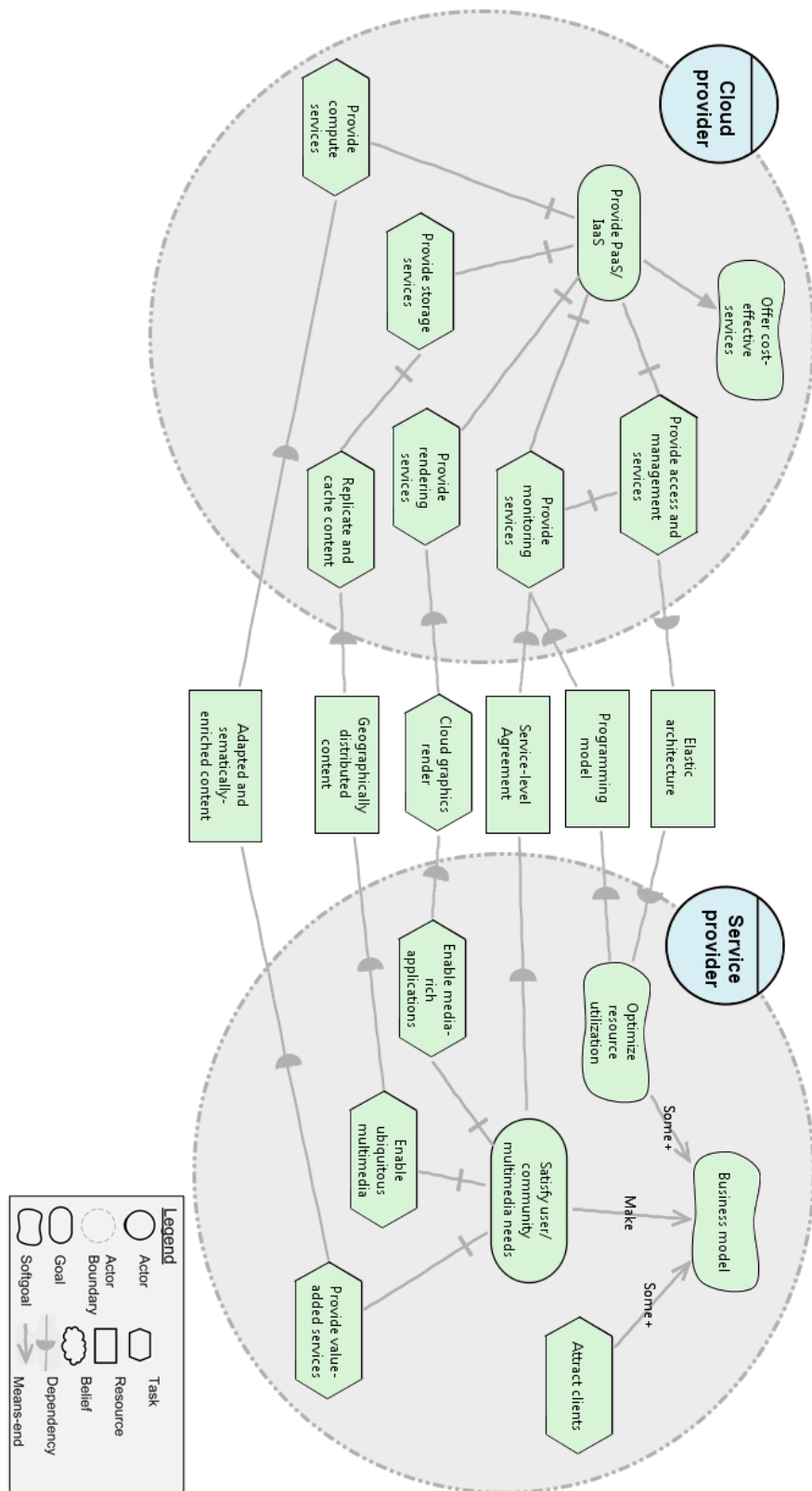


Figure 4.1: 1* Requirements model of multimedia services deployed on a "cloudified" server

An i^* model for multimedia services delivered as per “cloudified” server model is presented in Fig. 4.1. The main purpose of this model is to visualize the relationships between main actors in this architectural reference model and to characterize their intentions and express the overall complex requirements. The cloud provider and the service provider are modeled as agents (big dark circles) according to i^* terminology. The cloud provider presents a collection of diverse services which are provided as an IaaS/PaaS offer to service and application developers and providers. The cloud provider seeks to optimize the usage of own resources by using virtualization technology and multiplexing virtual computing machines on large hardware infrastructure (cf. Chapter 2). Cloud provider also aims to serve as many clients as possible to increase profits. On the other side, the service provider represents any application developer, company or organization which tries to satisfy in some end user or community needs in the form of multimedia services. Using their know-how and by renting cloud IaaS/PaaS, the service provider offers diverse applications and tools to end-users and community members. For example, rich-media applications (e.g. high-end 3D games) on resource-constrained mobile devices can be facilitated by using remote graphics rendering on cloud compute instances. Tools for ubiquitous multimedia production, processing, sharing, and distribution benefit from the geographically-distributed storage at the edge (cf. 2.3.2). Cloud compute instances can be scaled to provide (real-time) adaptation and semantic-enrichment of raw multimedia streams. In order to benefit from cloud IaaS/PaaS, the service provider needs to design a scalable software architecture based in the APIs and programming abstractions provided by the cloud provider, and at the same time it needs to optimize the resource usage and to keep costs low. The utilization of cloud services should adhere to the service level agreements for quality of service which at the end affects the overall user experience. All these three entities benefit from the cloud model. End users do not need to install applications or buy software licenses, service providers avoid making investment on servers, and cloud providers profit from IaaS/PaaS rental.

Multimedia applications considerably need support from cloud resources due to their compute-intensive and delay-sensitive requirements. In cloud terminology, compute-intensive requirements translate into resource costs. During service provisioning, if the

allocated cloud resources cannot satisfy the demand, on-demand resources can be rented instantly to meet the additional workload. However, service providers need to optimally allocate cloud resources for each service/application to achieve minimal resource cost and satisfy different QoS requirements. This is challenging because of the dynamic nature of different applications and resource demands. In a typical cloud environment, the service provider takes care of the optimal allocation of cloud resources by using the access and monitoring interfaces of a cloud IaaS/PaaS.

In many multimedia applications, the application responsiveness is sensitive to the round trip time (RTT). The RTT represents the sum of forward transmission delay from the end user device to the cloud provider, the backward transmission delay from the cloud provider to the end user device, and the service response time in the data center. For example, in interactive applications and games, a delay between user input and graphic output more than 100 ms affects user experience. Researchers and practitioners have proposed many methods on how to reduce the transmission delay within networks, e.g. adaptive and scalable video coding, caching, etc. However, these factors are immutable from service provider perspective who is able to work only with best-available resources. In this dissertation, I consider means to optimize the RTT and respective user experience from content and end user interaction perspectives. Section 6.1.2 deals with practical implementation of such optimizations.

MAPE-K Loop for Auto-adjustable Service Architecture

The different aspects of automatic management multimedia services on “cloudified” server architecture can be considered with help of the MAPE-K loop [IBM03] defined by IBM — a widely applied concept in automatic computing [CDD*12]. It defines control loops in a system to achieve self-configuring, self-healing, self-optimizing and self-protecting features, which are also very relevant in cloud-based systems. The MAPE-K loop consists of the following elements:

- **Monitor:** The monitoring element collects, aggregates, filters and reports the details about the managed resources. For example, it senses involved cloud resource usage at the different levels (hardware, IaaS, PaaS) and forwards this information to the analyze element.
- **Analyze:** The analyze function provides mechanisms to correlate and model complex details in order to determine if some change needs to be made. The system must be able to perform complex data analysis and reasoning, i.e. to make sense out of the stored shared data. For example, this function can determine that the multimedia system violates the service-level agreement and decides that change (now or in the future) in the system is needed to meet that requirement. If change is needed, a request is passed to the plan element which describes what modifications of the system components are deemed necessary or desirable.

- **Plan:** The plan element constructs the actions needed to achieve goals and objectives. The actions form a change plan which can contain from simple commands to complex workflows. For example, it can describe the desired set of changes for the managed cloud VM allocation to meet the current load.
- **Execute:** The execute element provides means to schedule and perform the intended changes to the system. It controls the execution of a plan with consideration of dynamic updates. The actions from the plan are executed through actuator interfaces of the system. For example, the cloud management API can be used to execute instantiation of new VMs.
- **Knowledge:** Data used by the other four elements (monitor, analyze, plan and execute) are stored as shared knowledge. This shared knowledge includes data such as historical execution logs and configurations, metrics, policies, etc. Knowledge from this data can also be inferred by using reasoning systems or machine learning techniques. For example, it is possible to express different goals of the system by defining utility functions.

Figure 4.2 depicts a model for automatic management of cloud managed resources for multimedia applications. The model relies on the MAPE-K loop philosophy with an additional inspiration from [KoBe12]. The automatic management of cloud resources depends on defining configurations at different levels. The configurations capture sets of parameters related to the resource requirements and the runtime of applications and services. A service configuration is composed of parameters that enable tuning of a service to specific hardware and execution goals such as responsiveness and cost constraints. For example, it can define the type of multimedia data that needs to be processed (image or video). The platform configuration characterizes the aspects such as scheduling policies and execution environment specifications. For example, in MapReduce environment, the platform configuration would define the number of map and reduce tasks per multimedia file or per user. The resource configuration captures the variable set of computational resources (CPU, GPU, hardware architecture, etc.). The job configuration integrates the other three configurations (service, platform and resource) in addition to a mapping to enable an adaptive configuration of execution jobs. For example, it can define how to split large video files to be processed in parallel or can define a global content replication and caching.

The left part of Figure 4.2 shows the main components of a “cloudified” server architecture for multimedia purposes. The functionality and semantics of these components are defined in Chapter 2. The right part of Figure 4.2 illustrates a MAPE-K loop. Diverse sensors for estimating the system state can be used, e.g. workload, resource usage, and pre-defined quality of service policies for multimedia delivery. Data and events from sensors at all layers feed the monitoring element which provides service, platform and resource configurations to the analysis element. The analysis element evaluates the actual configuration of all layers involved in the application execution. The findings from this element are forwarded to the

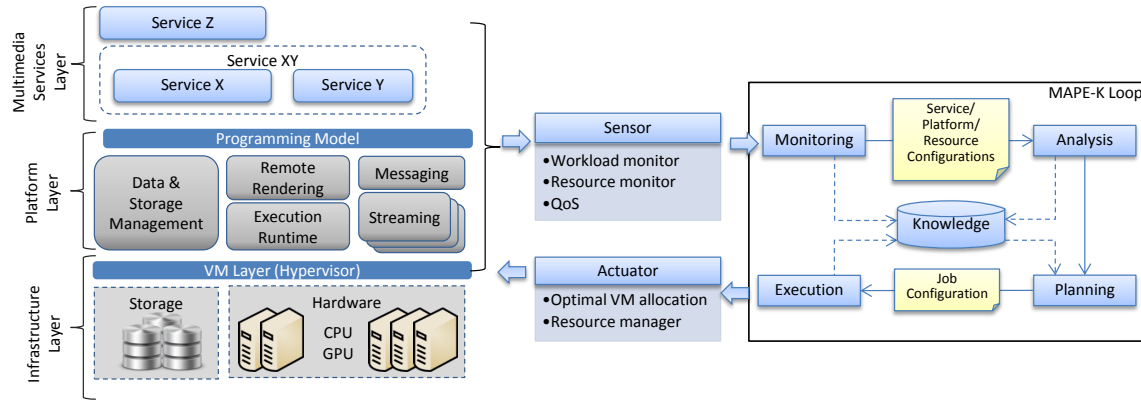


Figure 4.2: A cloud self-automated management model inspired by the MAPE-K loop

planning element and to the knowledge element to be used in subsequent decision processes. The planning element decides on job configurations that give highest utility values. These values are calculated with a utility function defined by some domain expert. In addition, the planning element uses a form of performance model to estimate the service/application execution. For example, a simple form of performance model is one based on historic execution logs. The execution element configures each layer according chosen configurations. For example, it can reserve cloud VM instances, configure them with multimedia processing libraries and initialize them for service execution. It also evaluates the execution and stores data about it in the shared knowledge element.

4.2.2 Cloud-supported Augmentation

The relative resource poverty of mobile devices as well as their lower trust and robustness lead to reliance to static servers [Saty96]. But the need to cope with unreliable wireless networks and limited power capacity argues for *self-reliance*. In fact, mobile cloud computing approaches must balance between these aspects. This balance must dynamically react on changes in the mobile environment. Mobile applications need to be adaptive, i.e. the responsibilities of client and server need to be adaptively reassigned.

The cloud computing concepts can be considered from a viewpoint of the mobile device. In fact, we want to achieve a virtually more powerful device, but in contrast to the previous model, we want to keep all the application logic and control on the device. There are many reasons why this kind of application model is desired. For instance, to retain privacy and security without sharing code and data with the cloud, or simply reuse the existing desktop applications on the mobile device, or reduce the communication latency introduced by the remote cloud.

There are many approaches that could be used to achieve mobile cloud integration. As discussed in Chapter 2 and Chapter 3, offloading has potential in mobile cloud computing,

since it supports cloud computing principles, i.e. it helps to surmount mobile devices' shortcomings by augmenting their capabilities with external resources. Furthermore, offloading is the basis for elastic mobile applications, i.e. applications that could seamlessly exploit both mobile and cloud resources, thus execution on a virtual execution environment richer beyond the device's physical capacities. Which portions of the application are executed remotely is decided at runtime based on resource availability. In contrast, client/server applications have static partitioning of code, data and business logic between the server and the client, which is done in development phase.

In order to dynamically shift the computation between a mobile device and a cloud, applications needed to be split in loosely-coupled modules (or services, components, etc. depending on the chosen granularity) which interact with each other. The modules should be able to dynamically instantiate on and shifted between mobile devices and cloud depending on the several metric parameters modeled in a *cost model*. The set of parameters can include module execution time, resource consumption, battery level, monetary costs, security and privacy policies, and/or network bandwidth. A key aspect is user waiting time, i.e. the time a user waits from invoking some actions on the device's interface until a desired output or exception is returned to the user. User wait time is important for deciding whether to do the module execution locally or remotely.

Figure 4.3 illustrates a diagram of the main activities in the offloading process during cloud-based augmentation of mobile devices. Whenever an application capable of offloading is invoked by the end user, the augmentation layer on the mobile device initiates a partitioning process. This process is closely dependent on the supported module encapsulation method and granularity. During partitioning, all offloadable parts of the application binaries are identified. This process can further be supported using application-specific partitioning data. For example, the developer can define explicitly which pieces of the application could be offloaded and which must remain on the device. Moreover, previous executions and partitioning decisions can be logged and associated with the real output. This could provide valuable knowledge for possible optimizations. Next, optimal constellations are chosen which satisfy certain goals and constraints. A partition decision consists of a classification between components to be offloaded to a remote cloud host and others to be executed locally. The remotely executed modules need to be registered at a chosen cloud provider which hosts an execution environment specific to the chosen encapsulation and virtualization technology. In addition, data which is required as input at the execution of the offloaded modules needs to be fetched and/or cached. At the end, the results of the remote execution is integrated with the local execution. However, the consistency and integrity of the performed execution needs to be verified. The result of the offloading must be identical with a pure local execution, but with some performance or energy-saving gains.

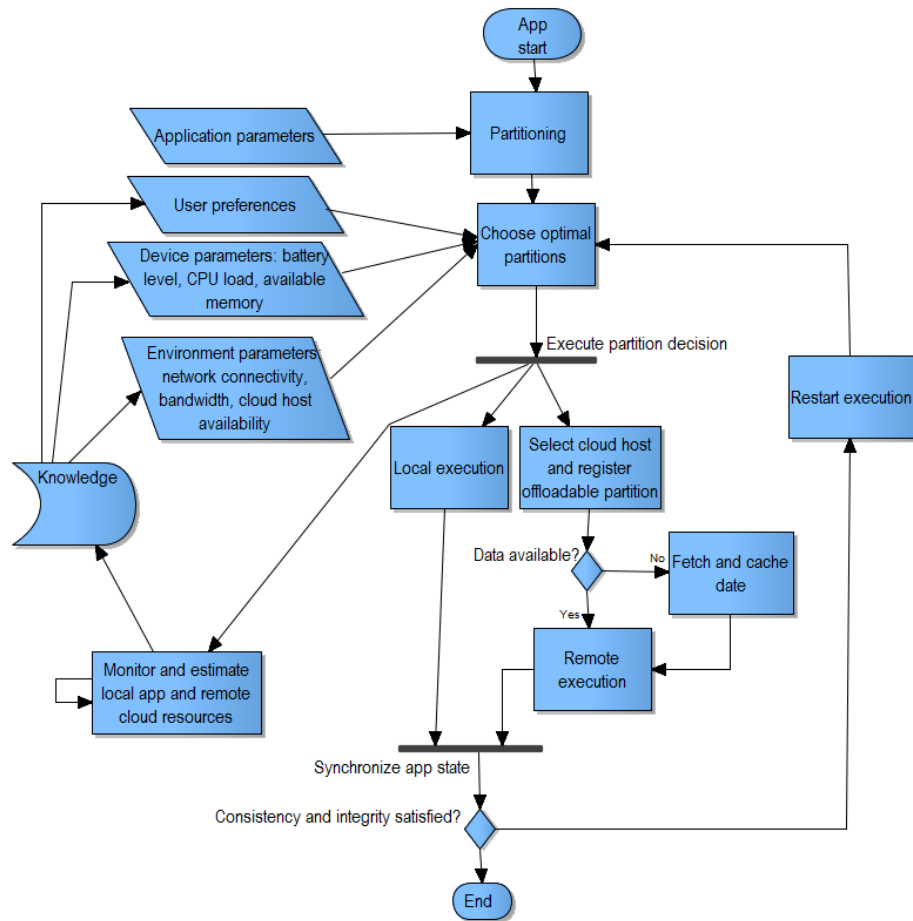


Figure 4.3: A diagram of the offloading process during cloud-based augmentation

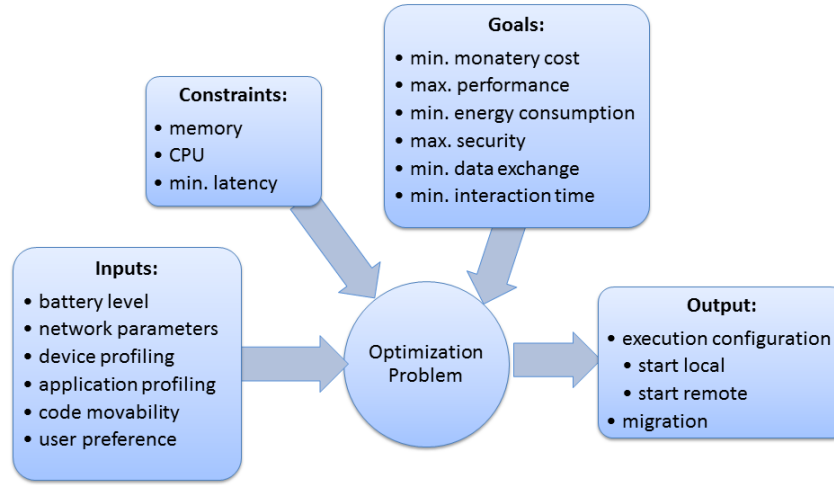


Figure 4.4: Cost model of elastic mobile cloud applications (extended from [ZJKG10])

Choosing Optimal Offloading Strategies

Which parts of the mobile application run on the device and which on the cloud can be determined based on a cost model. The cost model takes inputs from both device and cloud, and runs optimization algorithms to decide execution configuration of applications (cf. Fig. 4.4). For instance, Zhang et al. [ZJKG10] use Naïve Bayes classifiers to find the optimal execution configuration from all possible configurations with given CPU, memory and network consumption, user preferences, and log data from the application. Giurgiu et al. [GRJ*09] model the application behavior through a resource consumption graph. Every bundle or module composing the application has memory consumption, generated input and output traffic, and code size. Application's distribution between the server and phone is then optimized. The server is assumed to have infinite resources and the client has several resource constraints. The partitioning problem seeks to find an optimal cut in the graph satisfying an objective function and device's constraints. The *objective function* tries to minimize the interactions between the phone and the server, while taking into account the overhead of acquiring and installing the necessary bundles.

However, optimization involving many interrelated parameters in the cost model can be time or computation consuming, and even can override the cost savings. Therefore, approximate and fast optimization techniques involving prediction are needed. The model could predict costs of different partitioning configurations before running the application and deciding on the best one [ChMa10].

The proposed model and corresponding algorithm are supposed to be applied for scenarios which are computation-intensive [KPK*09]. Moreover, this model enables application and service developers to keep using their acquainted development approaches only with a minimal requirement that their code is structured in a modules/services.

Let us suppose that we have n number of modules which can be offloaded, $S_1, S_2 \dots S_n$.

Each of modules has several properties described as metadata, i.e. for specific module i , its memory cost mem_i , code size $code_i$. Let us consider the k number of related module which can be offloaded. For each of them, we denote the transfer size $tr_1, tr_2 \dots tr_k$, send size $send_1, send_2 \dots send_k$, receive size $rec_1, rec_2 \dots rec_k$, where $\{1..k\} \subseteq \{1..n\}$ and $send_k + rec_k = tr_k$. Meanwhile, we introduce x_i for module i , which indicates whether the module i is executed locally ($x_i = 0$) or remotely ($x_i = 1$). The solution $x_1, x_2 \dots x_n$ represents the required offloading partitioning of the application.

The cost function (i.e. utility function) is represented as follows:

$$\min_{x \in \{0,1\}} (c_{transfer} * w_{tr} + c_{memory} * w_{mem} + c_{CPU} * w_{CPU}) \quad (4.1)$$

where

$$c_{transfer} = \sum_{i=1}^n code_i * x_i + \sum_{i=1}^n \sum_{j=1}^k tr_j * (x_j XOR x_i) \quad (4.2)$$

$$c_{memory} = \sum_{i=1}^n mem_i * (1 - x_i) \quad (4.3)$$

$$c_{CPU} = \sum_{i=1}^n code_i * \alpha * (1 - x_i) \quad (4.4)$$

There are three parts in the cost function. The first part depicts the transfer cost for remote execution of services, including the transfer cost of its related services which are not at the same execution location. The latter part of Eq.(4.2) implicitly includes the dependency relationship between modules, i.e. if the output of one module is an input of another. The c_{memory} contains the memory cost on the mobile device, and c_{CPU} the CPU cost on the mobile device, where α is the convert factor mapping the relationship between code size and CPU instructions, which is taken to be 10 based on [Oust97]. w_{tr} , w_{mem} , w_{CPU} are the weights of each cost, which can lead to different objectives, for example lowest memory cost, lowest CPU load or lowest interaction latency.

The three constraints are expressed as the following.

Minimized memory usage. The memory cost of resident service can not be more than available memory on the mobile device, i.e.

$$\sum_{i=1}^n mem_i * (1 - x_i) \leq avail_{mem} * f_1 \quad (4.5)$$

where $avail_{mem}$ can be obtained from the mobile device, and f_1 is the factor to determine the memory threshold to be used (because the application can not occupy the whole free memory on the mobile device).

Minimized energy usage. For the offloaded services, the energy consumption of offloading should not be greater than not offloading [KuLu10], i.e.

$$E_{local} - E_{offload} > 0 \quad (4.6)$$

The local energy consumption can be expressed using the number of local instructions to be executed I_{local} , local execution speed S_{local} and the power used of local execution P_{local} [KuLu10]. At the first decision time, I_{local} is estimated according to the code size. After the first decision, this number is collected from our framework (by calling the statistic method provided from Android API). Obviously, there is a relationship between the instruction number and the power used for the respective instruction while performing power profiling.

$$E_{local} = \frac{P_{local} * I_{local}}{S_{local}} \quad (4.7)$$

The energy cost of offloading some parts to remote cloud can be expressed as the sum of energy consumption during waiting for the results from the cloud E_{idle} , transferring (including sending E_s and receiving E_r) the services to be offloaded [KuLu10] and also the additional data which may be needed on the remote cloud E_{extra} .

$$\begin{aligned} E_{offload} &= E_s + E_{idle} + E_r + E_{extra} \\ &= P_s * (t_s + t_{extra}) + P_{idle} * t_{idle} + P_r * t_r \end{aligned} \quad (4.8)$$

The idle time of the mobile device waiting for the result from cloud can be treated as the execution time of remote cloud, so the formula becomes

$$\begin{aligned} E_{local} - E_{offload} &= \frac{P_{local} * I_{local}}{S_{local}} - \frac{P_{local} * I_{local}}{S_{cloud}} \\ &\quad - \frac{P_s * (D_s + D_{extra})}{B_s} - \frac{P_r * D_r}{B_r} \end{aligned} \quad (4.9)$$

where D_s and D_r are the total data sizes to be sent and received, D_{extra} is the size of extra data needed because of offloading, which is determined at runtime, B_s and B_r are the bandwidths of sending and receiving data, and S_{cloud} is the remote execution speed.

Additionally,

$$I_{local} = \sum_{i=1}^n code_i * type_i * x_i \quad (4.10)$$

$$D_s = \sum_{i=1}^n send_i * type_i * x_i \quad (4.11)$$

$$D_r = \sum_{i=1}^n rec_i * type_i * x_i \quad (4.12)$$

where $type_i \in \{0, 1\}$ represents whether a service is offloadable or not.

Minimized execution time. The third constraint is enabled when the user prefers fast execution, i.e.

$$t_{local} - t_{offload} > 0 \quad (4.13)$$

The local execution time can be expressed as the ratio of CPU instructions to local CPU frequency, meanwhile, the remote execution time consists of the time consumed by CPU, file transmission and the overhead of our middleware.

$$t_{local} = \frac{I_{local}}{S_{local}} * x_i \quad (4.14)$$

$$t_{offload} = \left(\frac{I_{local}}{S_{cloud}} + \frac{D_{extra}}{B_s} + t_{overhead} \right) * x_i \quad (4.15)$$

where $t_{overhead}$ is the overhead which our framework brings in.

According to the constraints above, we now transform the partitioning problem to an optimization problem. The solution of $x_1, x_2, \dots, x_n$, is the optimized partitioning strategy. By using integer liner programming (ILP) on the mobile device, MACS gets a global optimization result. Whenever the parameters in the model change, e.g. available memory or network bandwidth, the partitioning is adapted by solving the new optimization problem.

4.2.3 Fog/Edge Computing

The previous two sections represent opposite computing models in the mobile cloud landscape. On the one hand, cloud services follow mostly a centralized architecture where the data and computations are located in cloud data centers. On the other hand, cloud-based augmentation provides full control to the mobile end user device and only opportunistically exploits cloud resources to host expensive operations. However, between these two opposites lies an opportunity that can combine the capabilities of both models. By placing compute, storage and networking services between end user devices and traditional cloud data centers, i.e. somewhere at the edge of the network, it is possible to overcome individual limitations. For example, having rich services near by the end users would facilitate better QoS in terms of delay and power consumption, reduction in data traffic from and to remote clouds, etc.

This idea has been explored in different research communities but named under different terms. Bonomi et al. [BMZA12] called it *Fog computing*, alluding to the fact that the fog is a cloud close to the ground. Reduced latency, media content and processing, aggregation are pushed at the edge of the network, i.e. mobile network base stations or WiFi hotspots. In a nutshell, fog computing offers combined virtualized resources such as computational power, storage capacity, and networking services at the edge of the networks, i.e. closer to the end-users. Fog computing supports applications and services that require very low latency, location awareness, and mobility. Fog computing complements the cloud services. Fog computing is a highly virtualized platform that provides compute, storage, and networking services between end devices and traditional cloud data centers, typically, but not exclusively located at the edge of network. Satyanarayanan et al. [SBCD09] define a similar concept called cloudlets, which are software/hardware architectural elements that exist on the convergence of mobile and cloud computing. They are the middle element in the 3 tier architecture — mobile device, cloudlet, cloud. Cloudlets emerge as enabling technology for resource-intensive but also latency sensitive mobile applications. The critical dependence on a distant cloud is replaced by dependence on a nearby cloudlet and best-effort synchronization with the distant cloud.

Cloudlets or fog computing nodes possess sufficient compute power to host resource-intensive tasks from multiple mobile devices. Moreover, they can enable collaboration features with very low latency, aggregation of stream data, local analytics, peer-to-peer multimedia streaming, etc. The end-to-end response times of applications within a cloudlet are fast and predictable. In addition, cloudlets feature good connectivity to large data-center-based clouds. Cloudlet resembles a “data-center-in-a-box” and a self-managing architecture to enable simplistic deployment at any place with Internet connectivity such as a local business office or a coffee shop.

Figure 4.5 illustrates the main idea behind fog computing. Mobile devices could access remote cloud providers, but could also leverage the idle nearby computing resources to augment their capabilities. Instead of accessing cloud services in remote data centers, mobile

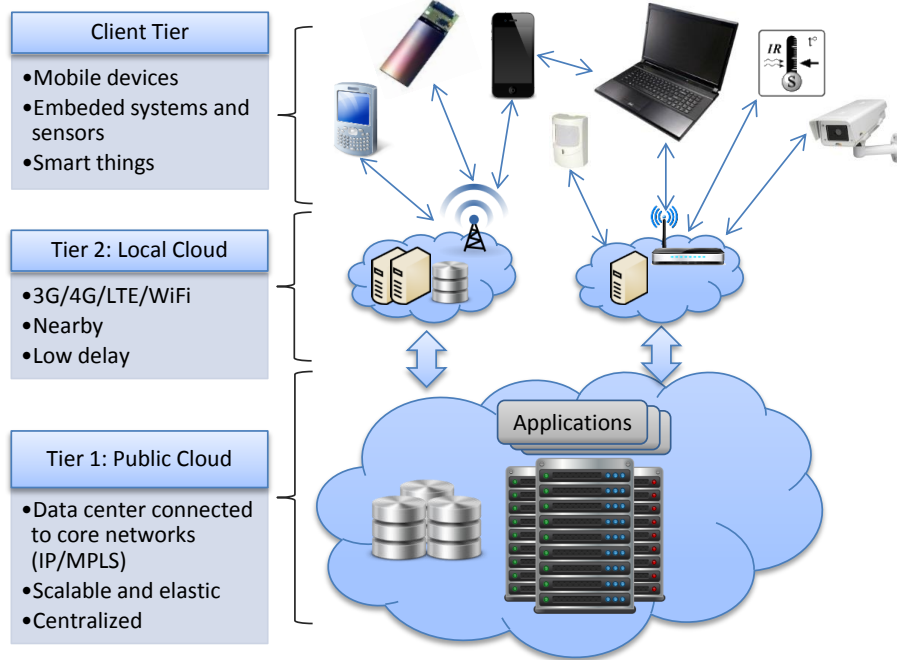


Figure 4.5: Simple architecture for Fog computing

clients could benefit from near by computing, storage and networking as envisioned by fog computing. The cloud tier is augmented with limited, local cloud-like resources which can be deployed at the edge of the network, e.g. cellular base stations or WiFi HotSpots. These so called cloudlets are only a few hops in the network to the end clients. This model for delivery of cloud services lacks a wide commercial adoption compared to the traditional cloud model, but it holds great potential. End users have access to more responsive cloud services, and cloud and network operators have relieved their core networks from data coming from millions of interconnected devices and sensors. The implementation of fog computing and cloudlets will follow the same cloud principles where compute virtualization and software-defined networking will be major forward-driving technologies.

4.3 Summary

In this chapter, three models of mobile cloud computing were identified, each with benefits and with drawbacks, which as a result affect the user experience, availability, responsiveness and provisioning costs of multimedia services. Furthermore, the issues and areas of concerns for multimedia services are grouped into three perspectives: system, mobile multimedia and user/community. The combination of these facets helps us to understand the main requirements for a platform that can serve mobile multimedia applications based on the cloud paradigm. All these three facets make a good structure how to take various complex

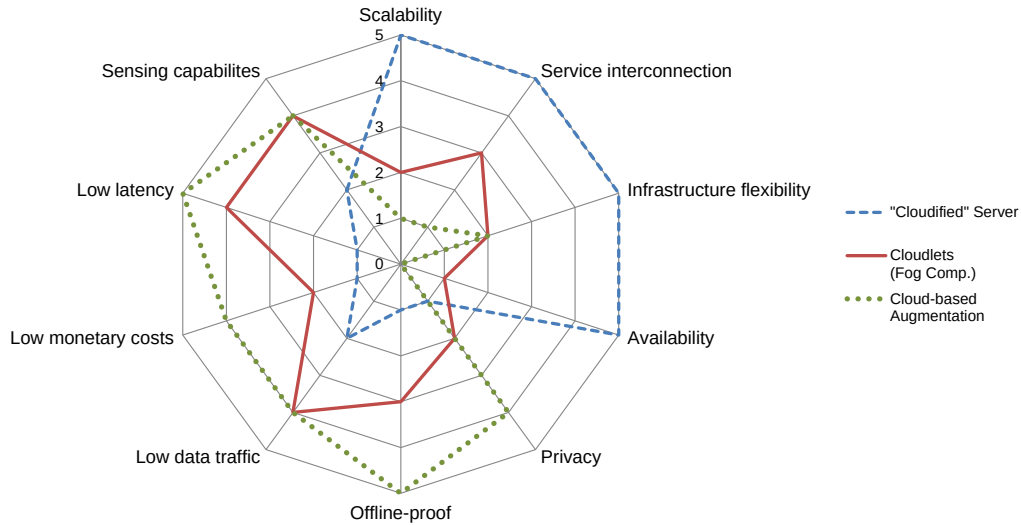


Figure 4.6: Feature comparison between the three mobile cloud computing models

Table 4.2: Mapping between the different classes of applications and optimal cloud models

"Cloudified" Server	Cloudlets (Fog Computing)	Cloud-based Augmentation
Social networking, big data, global availability	Local and field collaboration, low-latency and interactive apps, real-time streaming, Internet of Things, large-scale sensor networks	Offline-proof apps, privacy-preserving apps, natively-responsive apps

aspects at developing mobile multimedia cloud architecture into consideration. They have some overlapping sub-aspects as well, but the foci are distinctive. For example, from the technology facet, XMPP is related to provisioning of XMPP servers; whereas from the user and community facet it is related to realization of real-time collaboration. Furthermore, MPEG-7 and ontologies enable expressing multimedia semantics. The combination of MPEG-7 and XMPP enables easy creation of scalable semantic multimedia real-time collaborative applications. The facets, therefore, serve as an input in the designing the system architecture elaborated in the next chapter.

In addition, in this chapter, three cloud computing models are proposed which can to be taken as reference models. During the research and practical experiences, we realized that a single model will not suit all needs for mobile multimedia services. The diagram depicted on Figure 4.6 clearly justifies such reasoning. The three models are compared on the basis of several features relevant from the three-faceted view. Obviously, multimedia services using the traditional cloud model benefit from high scalability, globally-wide availability, easy interconnection to other cloud services and flexibility in creating custom application software architectures. On the other hand, such model lacks abilities to provide sensing capabilities, low response latency, low monetary costs and low data traffic. Here, the cloud-based augmentation approach solves some of the issues, also adding abilities of offline mode of application operation and inherent privacy. As discussed before, between these two

opposite models, opportunities for optimized features also exist. Fog computing models envision architectures that trade off between these two variants. This analysis could guide service developers in choosing an optimal model. Table 4.2 gives examples of which classes of applications suit better to which cloud model. Chapter 6 contains extensive examples of developed services and applications with evaluations to prove the validity of each model.

Perfect as the wing of a bird may be,
it will never enable the bird to fly if
unsupported by the air.

Ivan Pavlov (1849 - 1936)

Chapter 5

CAELUS: A Cloud Architecture for Enabling Mobile Multimedia Services

A primary task in supporting mobile multimedia services in the cloud is to offer the possibility to create scalable, elastic, application architectures with high performance and abstracted complexities. For that purpose the Cloud Architecture for Enabling Mobile Multimedia Services (CAELUS) is devised, which design and realization are described in this chapter. The architecture considers several actors (i.e. stakeholders) including cloud providers, content providers, service providers, end users and communities of practice. For example, service providers need development support in building applications that are usable and suit the needs of CoPs. The choices of a cloud platform have considerable implications on the design of mobile information systems. CAELUS facilitates the development, integration, authoring, administration and adjustment of mobile multimedia services which address different requirements. Using the general facets and mobile cloud reference models from the previous chapter, specialized requirements are deduced and further mapped to system components and services. This chapter introduces CAELUS architecture's core parts which include a test bed, offloading middleware and core multimedia services. The next chapter presents several mobile multimedia information systems built upon the core parts.

5.1 Key Requirements

The three faceted-view from Section 4.1 forms a general outlook of challenges and opportunities to realize mobile multimedia cloud computing. As it can be seen from Table 4.1, many research directions exist that go beyond the scope of this dissertation. However, the objective of my research is to examine the applicability of the emerging cloud paradigm in mobile multimedia services. A set of key requirements are derived in order to contain the research focus within the objective and research questions (i.e. research interests), but at the same time to encompass the aspects of the three facets. In fact, different research works tackle only parts of the whole. The study of existing solutions in Chapter 3 resulted that

none of them covers the three facets in a satisfying way. Moreover, the experiences with mobile multimedia community information systems influenced the identification of the key requirements.

This section discusses a set of key requirements that are crucial and minimal for the implementation of an appropriate mobile multimedia cloud architecture. In general, requirements define the functions of a system, constraints of its operations and specifications of system properties. These key requirements guide the design of the software prototypes in this work and the assessment of the approaches from related work. The key requirements for CAELUS include:

Adoption of the cloud computing paradigm: The whole architecture must follow cloud principles. Moreover, mobile constraints need to be included, too. Computing and storage functions need to be provisioned as a utility-like service model. The minimum criteria for this requirement is the ability of the architecture to elastically and automatically scale with the workload and assigned service policies. In addition, complex setup configurations of software and hardware must be hidden from the cloud service users.

Holistic service-oriented application architecture: The architecture needs to realize a systematic functional decomposition of multimedia applications into reusable cloud services. This requirement must be achieved through a holistic approach that covers end-to-end multimedia life cycle across heterogeneous mobile and Web platforms. The services should follow the service-oriented architecture principles and cloud delivery types.

Development support: The architecture needs to provide development support at different levels in order to embrace different levels of expertise, application flexibility or desired lead time. This requirement is accomplished by providing frameworks, core general services, programming abstraction and models.

Mobile and Web integration: The architecture must be designed around mobile devices from scratch. Mobile clients must not be taken as yet another additional end point, but as primary element around which the cloud services are build. Moreover, mobile clients are not isolated islands, but are interwoven in everyday activities of end-users. Thus, inevitably need to be integrated with the other Web services which also accessed through stationary clients.

Content and metadata management: The cloud platform of the architecture has to provide means for acquisition, sharing, transformation and delivery of multimedia content in various formats and over different networks. Equally important is to support mechanisms for enrichment, contextualization and adaptation of the content via metadata. The metadata needs to be accessible by other cloud services and needs to include information about user preferences and client computing environment.

These key requirements extract the essentials from the three facets that apply mostly for the cloud platform and related core mobile multimedia services in the CAELUS architecture. The first requirement follows directly from the objective of my research. The second requirement results from best practices and recommendations coming from the literature, but also from the experiences with our LAS community engine and its services. The third requirement implies directly from the main stakeholders of any software platform, i.e. developers. The fourth requirement embraces an ongoing trend in literature and practice of convergence between mobile and Web services. This was also identified during the usage of several Virtual Campfire prototypes (see Section 3.4). The fifth requirement follows naturally from the central digital artifacts considered in this dissertation. The importance of supporting both content and advanced metadata is elaborated through the whole dissertation starting from Section 2.5. Moreover, one can categorize the first three key requirements under the system and technology facet. The fourth and fifth requirements fit primarily under the mobile multimedia and slightly under the user and community facet. Of course, more specific requirements in diverse user and community domains exist, and they are elaborated in Chapter 6.

5.2 Design Considerations

The CAELUS ecosystem comprises a cloud platform and a variety of services which provide basic building blocks for complex mobile information systems. First, at the core of CAELUS architecture is *i5Cloud*, a cloud platform test bed for multimedia applications and services. *i5Cloud* is an implementation of the “cloudified” reference model (see Sec. 4.2.1). Second, CAELUS integrates an offloading middleware to enable elastic mobile applications according to the cloud-based augmentation model (see Sec. 4.2.2). To the augmentation scenarios, *i5Cloud* serves as a cloud host to offloaded partitions of mobile applications. Finally, CAELUS can be deployed not only on a standard large-size data center infrastructure (such as some public cloud provider) but also on a small-size private hardware infrastructure with limited resources, thus being able to serve as a cloudlet in fog computing scenarios (see Sec. 4.2.3). In short, CAELUS is able to emulate the three main reference models elaborated in the previous chapter.

The design of the *i5Cloud* test bed was followed by many design decisions. The following paragraphs give the rationale behind some of the decisions and the trade-offs made.

5.2.1 A Commercial Cloud Versus a Custom Test Bed

Generally, the aim of *i5Cloud* is not to compete with public cloud providers such as Amazon AWS or Google AppEngine, which provide a whole palette of generic compute and storage cloud services. In contrast, *i5Cloud* is a cloud test bed distinguished in several aspects. The role of *i5Cloud* is twofold. First, it is used to simulate a cloud environment where

we could have complete control over the infrastructure. This would not be possible if we have chosen to use a commercial cloud vendor despite some obvious benefits such as faster time-to-market options. Powerful hardware able to scale to a 128 virtual machines was already at our disposal at the outset of the research project. Other options such as the Eucalyptus project [NWG*09] existed, however, they were providing constrained solutions in terms of mobile and multimedia requirements. They were focused mainly on management tools for large-scale computing centers. Therefore, we opted for a custom cloud solution which considers the three facets and is able to facilitate various mobile cloud interplays.

The second aspect of i5Cloud is the ability to serve specialized mobile multimedia services. The idea was that we provide core services to manage multimedia content, metadata and context. In addition, service developers are enabled to create their own services. This could have been feasible by using again public cloud provider, but it would have limited us in the ability to have comprehensive insights “under the hood”. i5Cloud is a specialized platform that enables a single developer or a technical amateur to build large-scale multimedia applications. The burden of scalable multimedia and metadata management is leveraged by the platform. The application developer focuses on the application logic.

5.2.2 Public Versus Hybrid Cloud Strategy

i5Cloud features *hybrid cloud strategy*, i.e. i5Cloud takes advantage of in-house commodity hardware infrastructure which is usually available in most organizations, companies or institutions. In a case of cloud burst, i.e. when more resources are needed than those available in the private cloud pool, i5Cloud can automatically reach external public cloud infrastructures such as Amazon EC2. As a result of the hybrid cloud computing approach, we achieve a balance between resource limitations and re-utilization of existing infrastructure.

Intuitively, such hybrid cloud strategy has better security — main application components run on controlled hardware within the boundaries of an organization. Only limited data and code are deployed in a remote cloud data center. The critical pieces of the application reside on the private infrastructure. In addition, the hybrid approach exhibits lower latencies in terms of data locality. The private cloud resources usually reside near by the end users (community members) or service providers, thus high latencies to distant cloud data centers are avoided. In general, the hybrid strategy can be used by individuals or small organizations that want dynamically expand their system capacities by leasing resources from public clouds at a reasonable cost, but still retain control of own applications and data.

5.2.3 Cloud Interoperability

Adoption of cloud services relies largely at cloud interoperability and standardization. Every cloud provider has its own interface for interacting with the provisioned resources. However, this hinders cloud applications because of fears of vendor lock-in, non-portability, inability to mash up different cloud provider services, capturing of data and services in silos, etc.

While in my dissertation I did not aim to solve the issue of cloud interoperability, such a requirement arose from our hybrid cloud strategy. The hardware provided by public cloud vendors and our internal hardware differed on many levels including CPU architecture and operating systems. We solve that issue by adding a unified cloud interface which can support heterogeneous cloud resources. It abstracts the difference between various cloud APIs to provide a single common programmatic point acting as a cloud broker between remote platforms, networking, services, and data. This interoperability layer can be positioned at any cloud stack (infrastructure, platform, or services layer). In i5cloud we implemented one on infrastructure level because it results in better code reusability. In fact, platform components and multimedia services are executable on our private hardware and any public cloud that can feature the interoperability layer. Next sections give the technical details of the realization.

5.3 System Overview

5.3.1 The i5Cloud Platform Test bed

The purpose of any infrastructure is to offer an accessible collection of technologies that serve as foundation for other systems. Therefore, i5Cloud was designed to reduce technological complexity for different levels of expertise and flexibility needs of service providers. The i5Cloud test bed operates at three main layers: infrastructure, platform and multimedia. These layers, exposed to services providers via APIs, allow full access to the virtualized machines and storage, ability to compose desired applications out of existing components and mash up pre-defined services. The i5Cloud approach of offering several abstraction depths to trade-off complexity and flexibility achieves the *gentle-slope of complexity* [Beri04]. User need to be able to create small changes in a simple way, whereas more complicated changes should involve only incremental increase in complexity.

Since i5Cloud adopts the cloud paradigm, it needs to exhibit certain core cloud features. Referring to the requirements from Section 4.1, i5Cloud at the infrastructure level handles the management of large-scale data, adapts to variable workloads, supports dynamic configurations of hybrid cloud applications, etc. At platform level, i5Cloud employs multimedia specific libraries and complex configurations for various multimedia operations. Figure 5.1 gives an overview of i5Cloud. The virtualized computing and storage infrastructure enables scalable and highly-available multimedia-centric services with easy development.

In i5Cloud, three layers can be distinguished in the architecture diagram in Figure 5.1. From bottom up, the infrastructure and platform layer focus on requirements from the technology facet. The multimedia services layer considers the issues from the mobile multimedia facet. By using the multimedia services, developers can build scalable (mobile) multimedia applications that reflect the user and community requirements.

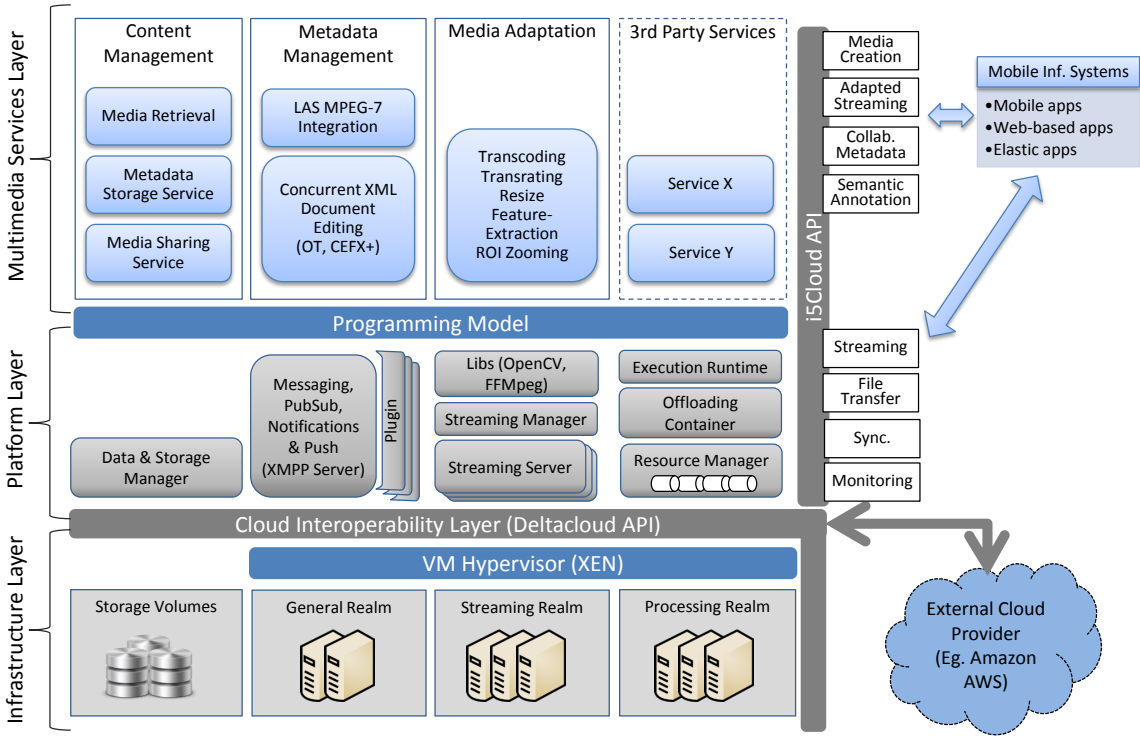


Figure 5.1: Layered architecture of i5Cloud

Infrastructure Layer

As described on many occasions throughout the dissertation, virtualization is the key technology for enabling cloud computing. In the case of i5Cloud, we embrace abstraction of the heterogeneous hardware through a *virtualization layer*. However, we go a step further by providing a *cloud interoperability layer*. i5Cloud uses Deltacloud API [DCApi] to enable cloud interoperability.

The interoperability layer based on DeltaCloud plays a big role in the i5Cloud architecture. Its RESTful API layer enables cross-cloud interoperability on infrastructure level with other cloud providers, e.g. Amazon EC2, Eucaliptus or GoGrid. The Deltacloud Core framework can be extended by creating intermediary drivers that interpret the Deltacloud RESTful API at the front end while communicating with cloud providers using their own native APIs on the back end. The drivers abstract the differences of different IaaS cloud providers. The Deltacloud API thus provides single unified access to heterogeneous cloud infrastructures.

Moreover, this cloud computing architecture is not constrained to run on our infrastructure only since the infrastructure layer runs on top of virtualized hardware. The upper three layers can easily be migrated on another public or private cloud infrastructure, thanks to the **virtualization abstraction on service level**. In other words, we are using an unified API for common cloud infrastructure management. This is crucial since many different virtualization layers exist. Popular cloud middleware like Eucaliptus or OpenNebula are restricted to the most popular virtualization technologies like Xen or KVM, which were not supported by our hardware. Deltacloud doesn't restrict the used virtualization technology. For example, our working prototype uses Sun Solaris Containers virtualization technology which is pre-built in our in-house hardware. Thus, Delatacloud makes it possible to use heterogeneous commodity hardware.

Virtual machines are grouped into realms which present boundaries between different computing resources. We have three realms, i.e. a processing realm for parallel and distributed computation over many machines, a streaming realm responsible for scalable handling of video streaming requests, and a general realm for running other servers such as XMPP or Web servers.

In our case, we use our in-house hardware supplemented with Amazon EC2 as a backup computing resource provider. The main parts of the API are:

- **Drivers:** Deltacloud drivers are the core feature of the API. They abstract the differences of different IaaS cloud providers. The Deltacloud API thus provides single unified access to heterogeneous cloud infrastructures.
- **Images:** Each image represents a virtual machine image in the back-end cloud, containing the root partition and initial storage for an instance operating system with pre-installed software.
- **Instances:** An instance is a running virtual machine. It is created from an image with preset hardware profile within a given realm.

- **Realms:** A realm is a logical boundary of a cloud resources. For example it can reflect an actual physical computing structure or a pool of related computing resources.
- **Storage volumes:** Storage volumes can be attached to a running instance and accessed by the instance's OS.
- **Buckets:** A bucket is the organizational unit of a blob storage. The blob storage represents a generic key/value data store.

The i5Cloud architecture provides web services to start, stop, persist, destroy and monitor virtual instances via the Deltacloud API. These virtual instances operate on dedicated CPU, virtual/physical memory and storage resources which are easily configured.

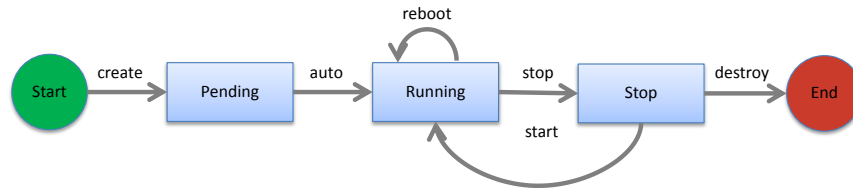


Figure 5.2: State machine diagram of cloud computing instances (according to Deltacloud API)

Platform Layer

Figure 5.1 illustrates the other main components of the i5Cloud test bed. The resource manager elastically scales up and down active computing nodes (virtual instances) according to the request demand. Different scheduling algorithms can be applied here to achieve the desired output. The resource manager monitors the complete workload and tries to optimally assign virtual machines according to the environment and job configurations. In the MAPE-K loop from Sec. 4.2.1, the resource monitor provides data to the monitoring component to some custom application, and also executes the actuator tasks.

In addition to computation and storage services, the i5Cloud platform also provides services for data streaming from the cloud to clients and vice versa. Media streaming is achieved by using standard software such as FFMpeg [FFMp13] and Wowza [Wowz12] servers, and text-based data (metadata) is streamed using XMPP servers [Real11b]. Thanks to the extensibility of XMPP, the OpenFire server can easily be extended with own or third-party plugins. Our collaborative metadata services demonstrate these features. The data storage manager handles the multimedia content and metadata in a highly-available manner. Some of the functionalities of i5Cloud at this level are exposed as platform services which the application developers can use for more flexible control of the execution environment. For example, file transfer and RTP streaming provide direct access to data storage. Furthermore, monitoring services could update applications with status information in real time. Execution

environments (e.g. Java runtime or application servers) are provided to services developers to deploy and execute their applications.

The upper layer of the architecture is explained in the forthcoming sections in more details. In brief, the concurrent editing and multimedia sharing components are the engine for the collaborative multimedia and semantic metadata services which are further the main building blocks for collaborative multimedia applications. The MPEG-7 metadata standard is employed to realize the semantic metadata services. Moreover, the intelligent media adaptation services enable more interactive mobile video applications, i.e. they contribute to a mobile user experience improvement.

5.3.2 Mobile Offloading Middleware

Additionally to i5Cloud we envision how to deploy mobile applications that are dynamically partitioned between limited mobile devices and the cloud with “unlimited” resources. To avoid the latency of accessing distant cloud centers and improve the user experience for resource-demanding applications, we consider the idea of offloading. Such a possibility opens the doorways to more powerful interactive mobile applications. In our group we developed a middleware for enabling mobile augmentation cloud services (MACS) with the goal to enable the execution of elastic mobile applications. The prototypical implementation and testing of MACS components were supported by a diploma thesis [Yu11].

Zhang et al. [ZJKG10] consider elastic applications to have two distinct properties. First, an elastic application execution is divided partially on the device and partially on the cloud. Second, this division is not static, but is dynamically adjusted during application runtime. The benefits of having such application model are that the mobile applications can still run independently on mobile platforms, but can also reach cloud resources on demand and availability. Thus, mobile applications are not limited by the constraints of the existing device capacities.

The architecture of our middleware prototype is show on Fig. 5.3. In order to use MACS middleware, the application should be structured using the established Android service pattern. An Android service is an application component that can perform long-running operations in the background and does not provide a user interface. For example, a service might handle network transactions, play music, perform file I/O, or interact with a content provider, all from the background. Android is already established as the most prominent mobile phone platform. Additionally, its application architecture model allow decomposition of applications into service components which can be shared between applications. A MACS application consists of an application core (Android activities, GUI, access to devices sensors) which can not be offloaded, and multiples services (S_i) that encapsulate separate application functionality (usually resource-demanding components) which can be offloaded (S_{Ri}). The services communicate with the application through an interface defined by the developer in the Android interface definition language (AIDL) [AIDL]. AIDL allows to define the programming interface that both the client and service agree upon in order

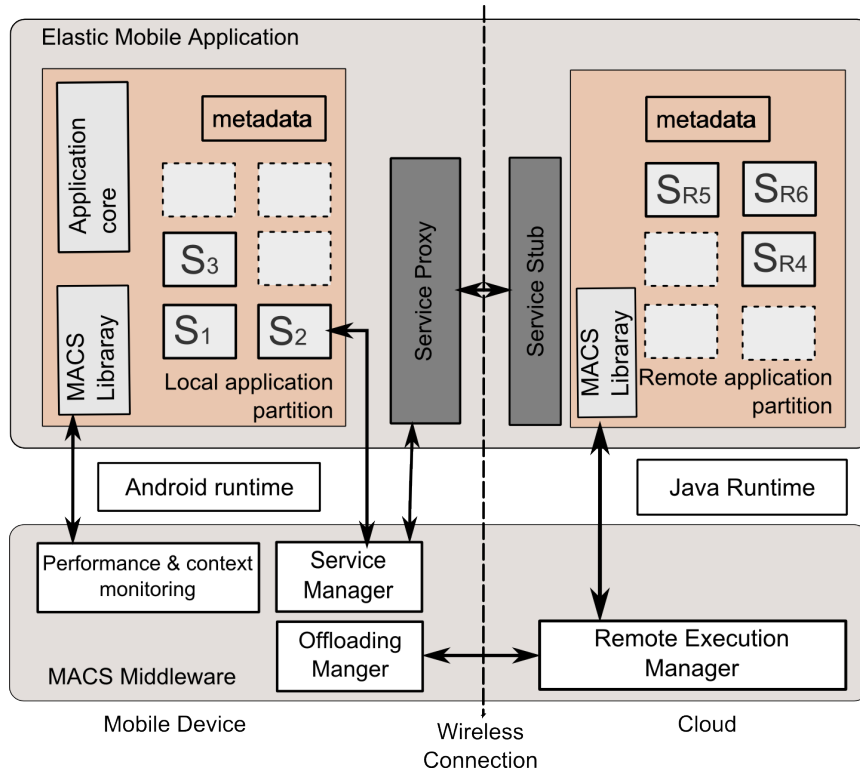


Figure 5.3: Mobile offloading middleware architecture

to communicate with each other using interprocess communication. AIDL handles the marshaling of objects into primitives for exchange between different processes.

As service-based implementation is adopted, for each service we can profile following metadata:

- Type: whether can be offloaded or not
- Memory cost: the memory consumption of the service on the mobile device
- Code size: size of compiled code of the service
- Dependency information on other services, for each related module, we collect following:
 - Transfer size: amount of data to be transferred
 - Send size: amount of data to be sent
 - Receive size: amount of data to be received

Metadata is obtained by monitoring the application execution and environment.

Android services are using Android interprocess communication (IPC) channels to do remote procedure calls (RPC). The services are registered in the Service Manager, and a binder maintains a handle for each service. Then an application, that wants to use a service, can query the service in the Service Manager. Upon service discovery, the Android platform will create a service proxy for the client application. All the requests to access the service will be sent through the service proxy, and then forwarded to the service by the binder. After processing the requests, results are sent back to the service proxy on the client application through the binder. Finally, the client gets the result from the service proxy. From client's point of view, there is no difference between calling a remote service or calling a local function.

The offload manager determines the execution plan, and then the services to be offloaded are transmitted to the cloud. The results are finally sent back to the activity. Our approach is similar to the Cuckoo framework [KPKB10], however, MACS allows dynamic application partitioning at runtime, where Cuckoo only enables static partitioning at compile time.

The main goal of MACS is to enable transparent computation offloading for mobile applications. Therefore, our middleware tries to fit the usual Android development process and bring developers an easier way to offload parts of their applications to remote clouds in a transparent way. MACS hooks into the Android compile system, makes modifications of generated Java files from AIDL in the pre-compile stage. Developers need to include MACS SDK libraries into their Android project.

At the cloud side, the MACS middleware handles the offload requests from the clients, installs offloaded services, their initialization and method invocation. The cloud-side MACS middleware is written with pure J2SE so that it can run on any machines with installed Java runtime environment.

MACS middleware monitors the resources on the mobile execution environment and available clouds. It then forms an optimization problem whose solution is used to decide whether the service which contains the called function should be offloaded or not. When the service is determined to be offloaded to the remote cloud, our middleware tries first to execute the service remotely. If there is no such service on the remote clouds, our framework transmits the service code to the cloud, and the corresponding results after the service execution are returned to the mobile device. The cloud caches the jar files for subsequent execution.

Except for the computation offloading, our framework also features simple data offloading. If files are needed to be accessed on the remote cloud, MACS file transmission (MACS-FTM) transfers automatically the non-existed files from the local device and vice versa.

5.3.3 Core Multimedia Services

This section covers some core services that leverage the i5Cloud infrastructure to mobile multimedia use cases which demonstrate the applicability and usefulness of our approach. First, we describe data management services for seamless mobile and Web integration in the

multimedia life cycle, i.e. acquisition, sharing, transformation and delivery of multimedia documents. Second, we explain services for multimedia metadata management used in the processes of enrichment, contextualization and adaptation.

Content Management

Mobile devices are becoming indivisible part of the Web today by a mobile and Web integration which not only means mobile Web pages, but also an integration of mobile devices as equal nodes on the Internet, same as desktop PCs and servers. Actually, the Web nowadays is a common communication channel for multimedia content that is “prosumed” on different personal computing devices such as desktops, laptops, tablets or smartphones. The ever increasing amounts of user-generated multimedia data require scalable data management in the clouds and ubiquitous delivery through the Web.

For example, let us consider cross-platform video data management as seen from the mobile multimedia facet and the technology facet. The example highlights the details of mobile and Web multimedia integration. Mobile applications seamlessly integrate with Web-based services. The seamless multimedia integration is done through i5Cloud which does the heavy-lifting of the necessary multimedia operations such as transcoding, adaptation, highly-available storage, responsive delivery and scalable processing resources. Figure 3.9 shows a screen snapshot of SeViAnno user interface where a user can navigate the video and place MPEG-7 based annotations on the video content, i.e. enrich it contextual information. This example demonstrates a current trend in Web applications, i.e. mobile/Web integration. Nowadays, most services are offered on different platforms using the Web as a common denominator. Seamless and uniform interaction with multimedia has become a prerequisite factor for successful services. However, the heterogeneity of end devices inevitably requires multimedia adaptation.

i5Cloud hosts core general-purpose multimedia-oriented services such those described in previous paragraphs. The requirements analysis has shown that content management services bring benefits to many multimedia applications. Service providers can reuse the common services in their applications, i.e. they are supported from reimplementing general multimedia functionalities.

Multimedia Acquisition and Delivery

Using mobile applications, users can record and annotate video and image content which can be uploaded or directly streamed to the cloud repository. In the i5Cloud, the videos are transcoded with a cloud video transcoding service. At the same time, the semantic metadata services handle the metadata content and store it in the MPEG-7 metadata store, making it available to other multimedia services. The management of all MPEG-7 data is implemented as LAS Web services [SKJR06]. The LAS multimedia and user management services enable Web clients to create, search, and retrieve MPEG-7 metadata. Afterwards, the uploaded videos are available in Web based applications for search, viewing and further

annotation. More details can be found in our research work [CRJ*10]. The transcoded video content is then accessible from various end client devices.

Multimedia Adaptation

Video media counts for the largest share in data transfer over the Internet. Additionally, video media is a very general term and includes concepts like codec, playback, streaming, recording among others. In comparison to TV and radio broadcasting every user is regarded separately and there exists a point-to-point connection between streaming server and the user's computer.

The overall user experience (UX) is one of the most important aspects for mobile video consumption [STWD10]. Several factors affect the user experience for mobile video, such as processing of videos, focusing/zooming on certain regions in the video, changes in connection speeds, browsing/navigating video on a mobile device, personalization of video streams. Furthermore, adaptation of multimedia resources has become an important task with the proliferation of end-user devices with different screen sizes and computing resources. Obviously, multimedia adaptation can play a big role in mobile multimedia services delivery.

For instance, possible solutions to overcome UX issues on mobile devices could include intelligent video processing like automatic feature extraction/object detection, zooming, bit-rate adaptation and automatic annotation [BZPr10, KPSV07, STWD10, BeGr09]. Above all, many trade-offs from the technology facet need to be considered since these intelligent video processing techniques cannot be provided by mobile devices, because they need a lot of resources and are very CPU intensive.

Therefore, the cloud computing paradigm is ideal to overcome these problems and to improve the user experience. The efficient use of available computational resources is very important to optimize the processing of high amount of data. Cloud computing is a key for this purpose with its “unlimited” computing and storage resources, processing power and parallel execution.

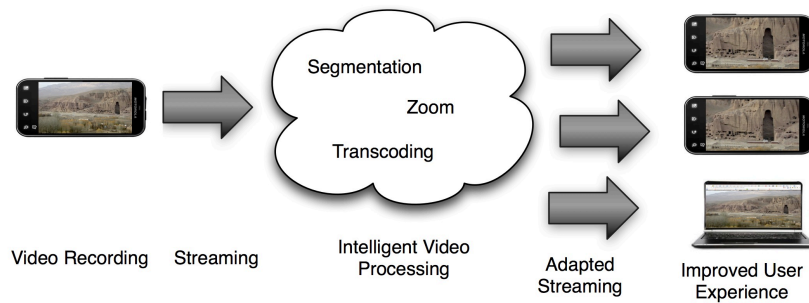


Figure 5.4: Intelligent and fast video adaptation cloud services

Figure 5.4 illustrates an example of a user who is recording events using a mobile phone's video camera (Fig. 5.4 left). The video is live streamed to i5Cloud. The cloud itself provides

various services for video adaptation such as transcoding and intelligent video processing services with feature extraction, automatic annotation and personalization of videos. This video is then directly streamed further and is available for watching by other users on mobile devices and Web clients (Fig. 5.4 right). Because of the intelligent video adaptation, users are able to watch videos with enhanced browsing supported by automatic annotations. Zooming additionally improves the overall user experience of video watching by focusing on the important parts of a video. Dealing with such complex set of adaptation services poses many research questions, however that is not the focus of i5Cloud. Its actual goal is to provide a scalable platform for such services.

Metadata Management

Metadata can be used to describe useful information about multimedia artifacts and its content in a machine-readable format. We provide a set of services for mobile clients to annotate multimedia collaboratively in real-time and to share multimedia and its metadata. Additionally, the services exploit the rich mobile context information.

Context-aware Semantic Annotation Services

The annotation is based on the MPEG-7 metadata standard [Kosc03]. MPEG-7 is one of the most complete existing standards for multimedia metadata. It is XML-based and consists of several components: systems, description definition language, visual, audio, multimedia description schemes, reference software, conformance testing, extraction and use of MPEG-7 descriptions, profiles and levels, schema definition, MPEG-7 profile schemas, and query format. However, several different approaches have been used as metadata formats in multimedia applications such as the Ontology for Media Resources [OMRe11] and COMM [ATS*07]. In order to enable interoperability between systems using these different formats, we have implemented or used mapping services. For example, our MPEG-7 to RDF convertor [CKKh09] is able to convert MPEG-7 documents into RDF documents for further reasoning and fact deriving about the multimedia.

Collaborative Metadata Services

We developed a mobile real-time collaborative system for multimedia that performs the collaboration on an open, customizable XMPP-based lightweight framework. It provides basic components for building mobile collaborative applications.

The user-generated multimedia content changes relatively slowly after its creation. On the other hand, the associated metadata is under constant modification. For example, a video creator initially describes and tags a new video. But after sharing the video, many other people contribute to the video with annotations, hyperlinks, comments, ratings, etc. Therefore, the success of multimedia services highly depends on features for metadata sharing collaborative metadata editing.

One of the key services for real-time collaboration is shared editing of XML documents. XML has been established as de facto standard for data exchange and interoperability, in-

cluding multimedia metadata standards. Since the nature of XML is generic and extendable, different kinds of information can be represented, such as graphic files (SVG), augmented reality content (ARML), multimedia metadata (MPEG-7), etc. In practice, real-time collaboration within multimedia applications generally means concurrent editing of XML metadata documents.

In short, collaborative services become increasingly important in many multimedia applications. i5Cloud, therefore, implements services for collaboration using multimedia content and metadata as central shared artifacts.

5.4 Summary

This chapter described and introduced the core parts of the CAELUS architecture which are relevant for the provisioning of complex mobile multimedia services. This CAELUS architecture addresses various mobile multimedia requirements identified in Chapter 4 and then further refined at the beginning of this chapter.

Table 5.1 illustrates a detailed comparison of CAELUS with some related works. Recent and previous related research work has largely focused on MapReduce-based [DeGh08] cloud solutions for multimedia application. For example, in [Sand11] is shown how to transform images and videos into cartoons or in [GKFu10] is demonstrated how to transcode media content into various video formats for different devices and different Internet connections. In all these approaches a video file is split in multiple parts, processed using parallel computing and merged in the correct order. Pereira et al. [PABE10] also propose an approach of using MapReduce for video transcoding. Their approach is to split the video into chunks and process the chunks in parallel using MapReduce jobs. Furthermore, Chen and Schlosser [ChSc08] propose to use MapReduce to speed up image processing in special feature detection. This shows that MapReduce can be used not only to speed up transcoding but also for feature detection in videos as video frames can be used as image input. White et al. [WYLD10] and Chen and Schlosser [ChSc08] go a step further to provide multimedia development support and tools for computer vision based analysis of multimedia on cloud platforms. Cloud-based augmentation approaches such as CloneCloud [ChMa09] and MAUI [CBC*10] solve the issue of mobile and cloud integration, but fail to fulfill the other requirements. On the other hand, approaches such as Song et al. [STWD10] give good solutions to multimedia-related issues but lack of cloud integration.

In short, most of the related systems do not provide a mobile multimedia platform for diverse services. They focus on cloud computing platform processing capacities and just pre-process multimedia files or use the cloud infrastructure just for storage and delivery. This is insufficient for enhanced overall mobile multimedia user experience as summarized by the requirements, e.g. end users also want to share content and need near real-time delivery and synchronization. Our mobile cloud services are proposed to simplify the realization of complex multimedia tasks. A contribution of the multimedia services layer

Table 5.1: Comparison of CAELUS with related approaches

	KR1	KR2	KR3	KR4	KR5
VideoToon [Sand11], Garcia et al. [GKFu10], Pereira et al. [PABE10], etc.	MapReduce-based	No	No	No	No
White et al. [WYLD10], Chen and Schlosser [ChSc08]	MapReduce-based	No	Dev. libraries	No	Computer vision algorithms for metadata enrichment
ChunkStream [PaPh10]	No	No	Framework for split videos	Balanced data between device and cloud	Content delivery only
Summa et al. [SVPS11]	Yes (MapReduce and cluster)	No	Algorithm support	No	No
CloneCloud [ChMa09], MAUI [CBC*10], etc.	Cloud-based augmentation	No	No	Offloaded code and data	No
Cloudlets [SBCD09]	Fog computing approach	No	No	Mediator between devices and clouds	No
Song et al. [STWD10]	No	No	No	Support for different devices	UX optimizations
CAELUS & i5Cloud	Parallel processing, scalable hybrid cloud architecture	Core, advanced and 3rd party services	PaaS & offloading support	Elastic apps, real-time data synchronization, messaging, etc.	Various content and metadata services

Legend: KR1: Adoption of the cloud computing paradigm
KR2: Holistic service-oriented application architecture
KR3: Development support
KR4: Mobile and Web integration
KR5: Content and metadata management

of i5Cloud and client applications is to augment resource-poor mobile devices through cloud services such as fast intelligent video adaptation services, in order to enhance user experiences in mobile multimedia and to enable metadata creation, real-time sharing, and concurrent collaborative editing.

The CAELUS approach helps service providers to change their development practices by providing means to design and deploy large-scale multimedia applications with less efforts. i5Cloud and the core multimedia services realize many of the considerations from the system and mobile multimedia facets 4.1. This chapter provided insights and solution ideas in the scope of my first research question (see 1.2). In the next chapter, the design and system architecture are tested in several domains to validate the CAELUS approach.

Facts are the enemy of truth.

Miguel de Cervantes (1547 - 1616)

Chapter 6

CAELUS-based Mobile Information Systems

The CAELUS architecture presented in Chapter 5 drafted a cloud multimedia platform considering the needs of service providers in the software engineering process. This chapter describes the assessment of CAELUS architecture from the three facets described in Chapter 4. i5Cloud platform and its core services form a strong basis for the design, implementation and deployment of advanced mobile multimedia services and information systems. They are presented here as a proof of validity and applicability of the conceptual framework and software architecture. CIViTra is a cloud video transcoding service based on i5Cloud. CIViTra together with a collection of mobile video and mobile augmentation services (MVCS and MACS) characterize the cloud system properties of CAELUS architecture. Then, mobile and Web integration is demonstrated with the mobile multimedia-centric information systems such as AnViAnno and XMMC. They are evaluated in the professional domain of cultural heritage documentation. The ubiquitous multimedia management and collaborative metadata services were assessed from community facet perspective. Furthermore, two user-centric cloud information systems applied in the domains of technology-advanced learning and human-computer interaction are described. At the end, the chapter summarizes the results from the use case studies.

6.1 Cloud Platform

This section covers the CIViTra, MVCS and MACS collection of cloud services. Basically, within the context of this chapter, they fall in the scope of system and mobile multimedia requirements facet, and try to fulfill the first three key requirements (KR1, KR2 and KR3). However, their prototypical implementation goes beyond that. For example, MVCS evaluates the usability of mobile transcoding services from user-subjective perspective. MACS serves as a complete cloud-based augmentation middleware.

6.1.1 CIViTra: Cloud Video Transcoder

Cloud Video Transcoder is a scalable hybrid cloud application which uses i5Cloud and Amazon Web Services. Video transcoding [VCHu03] is a CPU intensive operation, therefore, it is a suitable test domain for scalable cloud applications. The main idea behind the Cloud Video Transcoder is to use own private cloud infrastructure, and to start and use extra instances from a public cloud provider if demanded. Concretely, users upload multiple videos to the system, and if the number of videos in the transcoding queue is more than the number of free instances in the cloud, new instances are started and the videos are transcoded in parallel. When the transcoding queue empties, the extra instances are terminated. In the cloud, the cloud video transcoding service transcodes the video into streamable formats and stores different versions of a video.

The i5Cloud runs on a Sun SPARC Enterprise T5240 Server with Solaris 10 operating system. The server supports maximum 128 simultaneous threads, that means 128 single-threaded instances as virtual machines. Amazon EC2 is used as external public cloud provider. Table 6.1 contains information about the hardware profiles of the used VM instances (note: One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor.) The i5Cloud instance types are modeled according Amazon's setup, in order to have coarse comparison at least. FFMpeg version 0.8.2 is used as a video processing library.

The usage of computing resources of i5Cloud can easily be monitored in real time. The resource manager exposes its status through the "monitoring" interface (i5Cloud API). The resource monitor depicted on Figures 6.1, 6.2, 6.3 shows the status monitoring of resource usage during video transcoding tasks from different original formats into selected formats. Additionally, the cloud resource allocation can be visualized in real time during transcoding operations of uploaded videos. The overall CPU workload is balanced and reduced through employment of additional Amazon EC2 instances. They have been started based on a scheduling algorithm, in order to support i5Cloud during a request peak (see Figure 6.3). More cloud instances can be instantly employed depending on the workload. When the workload lowers, the number of instances is reduced to optimize resource usage for efficiency.

Figure 6.4 shows a comparison chart of transcoding execution time on different compute instance types for different file sizes. The three videos were with identical parameters, i.e. same codec, bitrate, and frames per second.

Figure 6.5 clearly shows the advantage of the cloud computing approach. Using commodity hardware and virtualization technology enables us to get a hybrid cloud computing environment that elastically scales up and down depending on the workload. Particularly, the chart plots the execution times for transcoding of 15 identical videos. Each video was 10.73 MB, with duration of 2 min. 8 sec. and encoded with a wmv3 codec. The videos had bitrate of 626 kb/s and 30 frames per second. The number of videos was kept constant and only the number of instances was increased at every iteration starting with 2 instances till 16



Cloud Video Transcoder

[Home](#)
[Upload](#)
[Gallery](#)
[Monitor](#)

Video File	Size (MB)	Duration	Codec	Instance	Start time	End time	Status
DLP_Part2.wmv	10.73	02:08	wmv3 (Main WMV3)	i-c84e5ebe	Wed Sep 07 16:09:48	Wed Sep 07 16:15:24	Transcoded
neverthere.mpg	27.24	02:43	mpeg1video	i-4e7e6e38	Wed Sep 07 16:14:32		Processing frame=176 time=00:00:02.66
Comic.avi	9.86	00:35	msmpeg4v2	i5c-32	Wed Sep 07 16:14:49		Processing frame=236 time=00:00:04.83
thehelp.mov	52.06			i5c-33	Wed Sep 07 16:14:47		Processing
Jet.avi	8.99	00:33	mpeg4 (DX50)	i-c84e5ebe	Wed Sep 07 16:15:30		Processing frame=436 time=00:00:11.34
yougot thepower.flv	30.37						Initial
AdaKrawcyk.flv	28.38	04:09	flv	i5c-34	Wed Sep 07 16:09:09		Processing frame=5790 time=00:01:54.17
searchspok.avi	9.71	01:29	msmpeg4v1	i-c84e5ebe	Wed Sep 07 16:12:55	Wed Sep 07 16:15:24	Transcoded
FPS_game.wmv	34.09						Initial

Figure 6.1: Cloud resource usage monitoring console: a list with details about videos being processed

Instance Id	Public DNS	Provider	Hardware profile	Launch time	Status
i5c-32	cloud32.dbis.rwth-aachen.de	i5Cloud	m1.medium	Wed Sep 07 16:19:41	Busy
i5c-32	cloud32.dbis.rwth-aachen.de	i5Cloud	m1.medium	Wed Sep 07 16:19:46	Busy
i5c-32	cloud32.dbis.rwth-aachen.de	i5Cloud	m1.medium	Wed Sep 07 15:52:30	Busy
i-4e7e6e38	ec2-46-137-129-13.eu-west-1.compute.amazonaws.com	EC2	m1.small	Wed Sep 07 16:17:41	Busy
i-c84e5ebe	ec2-46-51-157-87.eu-west-1.compute.amazonaws.com	EC2	m1.small	Wed Sep 07 15:10:47	Busy

Figure 6.2: Cloud resource usage monitoring console: status of the processing instances

Table 6.1: Hardware profile details of the i5Cloud and Amazon EC2 instances

	i5Cloud			Amazon EC2		
	micro	small	medium	micro	small	medium
Memory	1 GB	2 GB	2 GB	613 MB	1.7 GB	1.7 GB
CPU	2 CPU threads	4 CPU threads	8 CPU threads	Up to 2 EC2 Compute Units (for short periodic bursts)	1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)	5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)
Instance startup time	96s	96s	97s	234s	232s	235s
Max. No. instances	50	25	12	∞	∞	∞

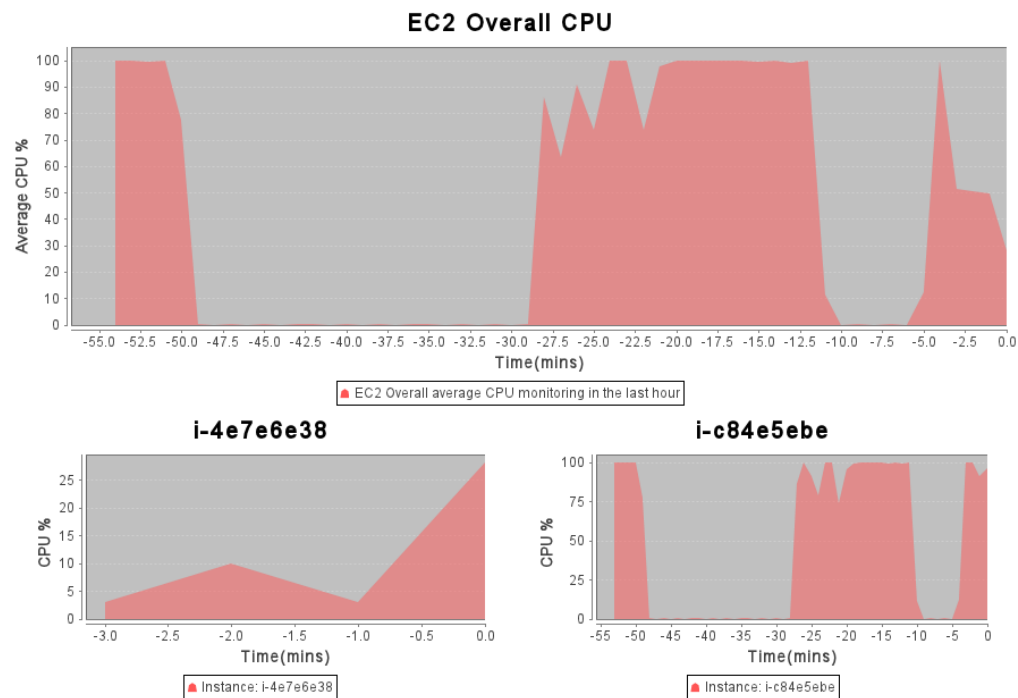


Figure 6.3: Cloud resource usage monitoring console: CPU load of processing instances

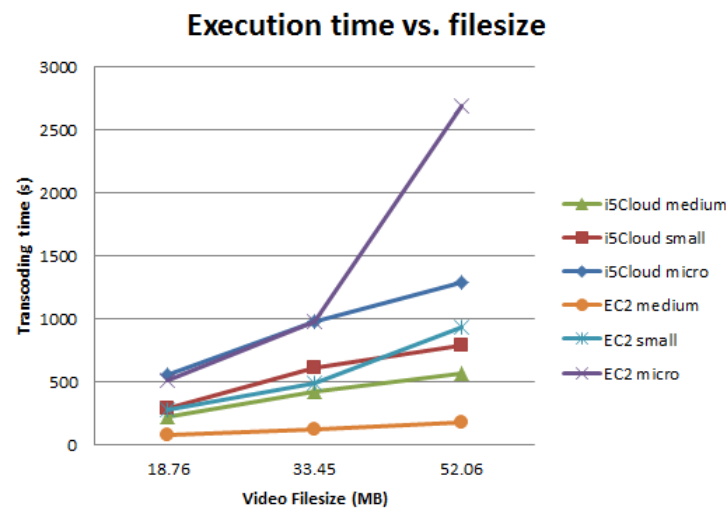


Figure 6.4: Transcoding execution time: iSCloud private cloud and Amazon EC2

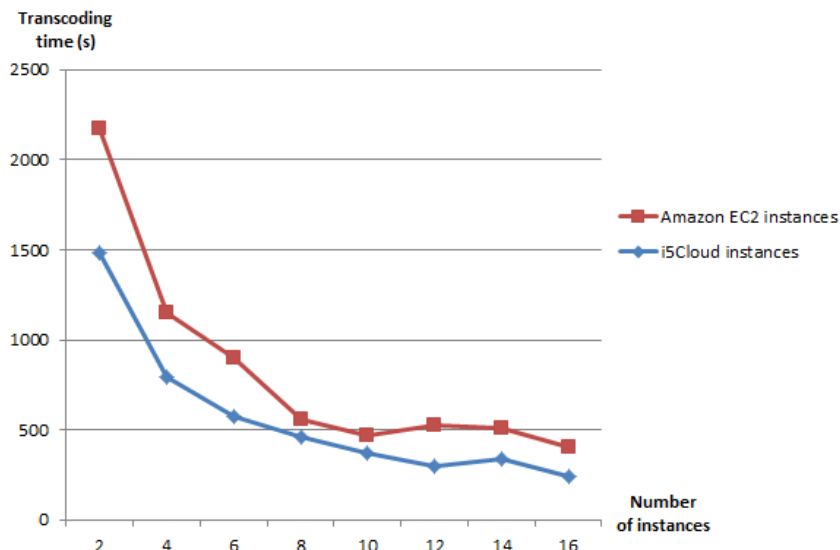


Figure 6.5: Scalability of i5Cloud

instances. Further increase of the number of instances would not affect the execution time since the number of instances would be greater than the number of videos, i.e. only 15 instances would have been busy and the rest would have been idle. The diagram clearly shows the scalability features of our i5Cloud architecture, i.e. the execution time can be reduced with increase of the number of instances. The scalability performs similar on our private infrastructure and on a public cloud infrastructure (Amazon EC2). Please note that this diagram doesn't serve to show that i5Cloud private instances perform better than Amazon EC2, but that they exhibit a similar scalability feature. The differences in execution times are probably due to the different hardware profiles.

Scaling Simulation

The goal of i5Cloud (and Cloud Video Transcoder) is not be a data center, but rather using cloud computing principles, to provide development primitives for multimedia applications, which common single-machine applications cannot provide. Additionally, the advantage of multimedia applications developed on top of i5Cloud is that they can easily be deployed on other cloud infrastructure. Therefore, two factors in the this use case are important, i.e. the ability to scale with the load and offload to other infrastructure when necessary, which are shown above.

Nevertheless, real-world web applications need to handle thousands of streams. In order to show the benefit of this hybrid approach, we conducted a simulation using the CloudSim toolkit which is one of the most popular cloud simulation tool [CRB*11]. CloudSim allows modeling of datacenters, physical hosts, virtual machines, processing jobs, etc. We modeled the Cloud Video Transcoder with two datacenters representing the private i5Cloud and

the public Amazon cloud. The parameters for the virtual machines were set the same as the small instances at i5Cloud and Amazon. The other parameters were inferred from the empirical results presented above. The experiment consisted of simulating sudden request demand of 10000 videos for processing. Each video was set to have size of 100MB. The simulation was repeated five times with the maximum number of private (i5Cloud) instances (25), and then adding 5 additional Amazon EC2 instances at every iteration. Figure 6.6 shows the speed up in processing by simply offloading in the public cloud and respective cost by using Amazon provider¹. For the sake of simplicity, in this model the network latencies were omitted in the scheduling algorithm and the data transfer costs were ignored. These results are very coarse estimate. The maximum 25 i5Cloud medium instances are running all the time. Upon a requests spike of 10000 videos, the Cloud Video Transcoder offloads to the Amazon cloud. At each iteration 5 additional EC2 instances are added, which reduces the total processing time, but on the other hand it increases the monetary costs.

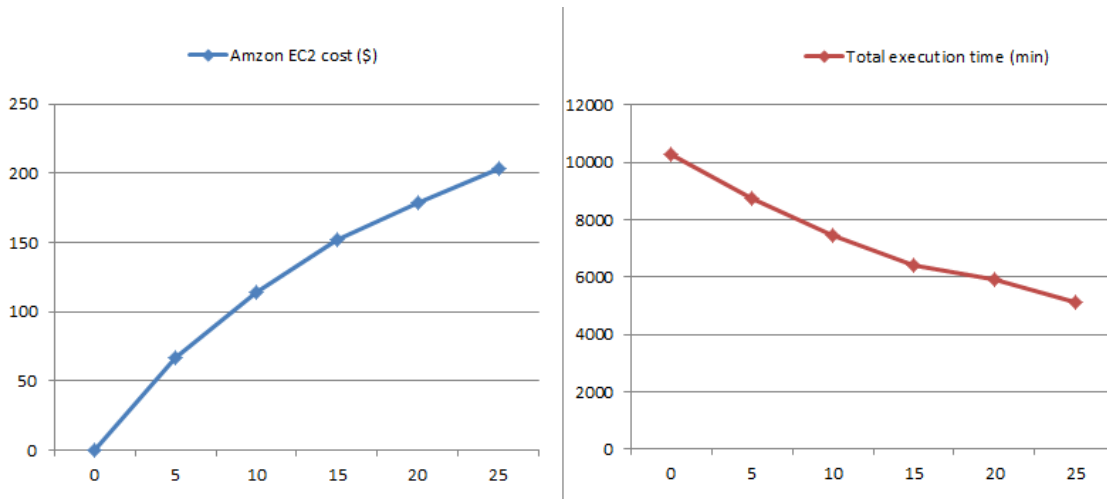


Figure 6.6: CloudSim simulation of Cloud Video Transcoder

6.1.2 MVCS: Improvement of User Experience for Mobile Video

One of the most important aspect for mobile video viewing is the overall user experience (UX) [STWD10]. We seek to remedy some of the issues with UX in mobile video applications by making use of cloud services for fast and intelligent video processing. The cloud services are also open to third party developers. They can easily extend/add custom adaptation functionalities to the video content/streams to improve the mobile UX in their apps. We developed our Mobile Video Cloud Services (MVCS) system for video processing in the cloud, where the prototypical implementation and testing of MVCS components were supported by a master thesis [Lott12]. In summary, we provide an open extensible cloud

¹<http://aws.amazon.com/ec2/pricing/>

platform for distributed video processing and an evaluation of the platform in improvement of mobile UX.

Several problems affect the mobile user experience during video delivery, including:

- The highly growing success of these mobile services leads to a high amount of multimedia data which needs to be indexed, summarized and to be adjusted to improve the user experience. Processing a huge amount of data like videos is very CPU intensive and needs a lot of data storage.
- Recognizing certain aspects in professional videos on a small screen is often very hard. Professional videos are mostly produced for the big screen and cover a panoramic view. Let's take a football game for instance. The distance of the camera to the football field is quite high. Therefore, it is very hard to recognize the current position of the ball.
- Rapidly changing and low bandwidth results in artifacts in the video. Due to the different covering of mobile networks the bandwidth can change rapidly during movement especially in cars or trains. Therefore, mobile users may enter areas of low bandwidth several times while watching a video. Low bandwidth can result in artifacts in the video and result in bad experience. Objects and actions may be unrecognizable.
- Browsing a video can be very tedious. Having a small screen size and a video with a long duration results in a seek bar with low accuracy. You will need to guess the desired position. Approaches like adjusting the speed of seeking dependent on the distance of the finger to the seek bar are helping to overcome this problem but do not solve it completely. It does not solve the problem of browsing a streamed video for bad Internet connections as the video player will only show you a black screen. The only indication of your current position is the time point which is not helpful if you do not know the content of the video.
- Watching several video streams at the same time is another problem. Again because of the limited screen size it is not possible to watch several videos at the same time. For example of the same event but from different point of views.

Video zooming, segmentation, and event/object detection are often proposed techniques for video retargeting to mobile devices. For example, zooming and panning to the regions of interest within the spatial display dimension can be utilized. This kind of zooming displays the cropped region of interest (ROI) at a higher resolution, i.e. observe more details. Panning enables watching the same level of zoom (size of ROI), but with other ROI coordinates. For example, in soccer game this would mean watching how player dribble with the ball more closely, whereas by panning one can observe other players during the game such as the goalkeeper.

MVCS Workflow

Figure 6.7 shows a workflow scenario of MVCS used to deal with the problem of the small screen size and video browsing. A mobile app developer can use different devices to record a video (1). Depending on the actual internet connectivity, the MVCS-enabled mobile client application can immediately stream the video to the cloud (2.1) or upload it later (2.2). The cloud transcodes the video into a common format for heterogeneous end devices (3.1), stores it on a streaming server (3.2). At the same time, the intelligent video processing algorithms are applied which can help improve the mobile user experience. For example, the video is split into segments based on a scene detector (3.3). A thumbnail for each segment is created. Object recognition, face recognition or motion saliency detection (3.4) are applied to the video stream in order to detect the regions of interest. These are just some examples of possible computer vision algorithms that can be used. Third party developers can apply their own algorithms depending on their application requirements. The gathered information is used for adjusted zooming level during the streaming of the video (4).

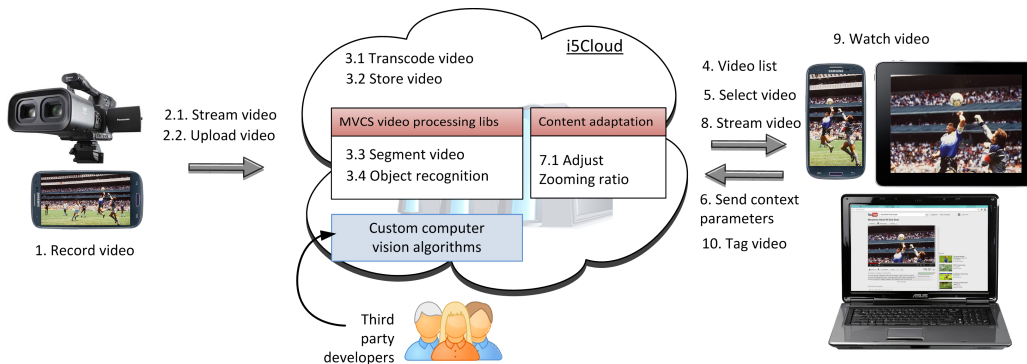


Figure 6.7: MVCS scenario workflow

End users can choose a video stream to watch (5). Based on parameters like screen-size, current bandwidth or selection area transmitted to the cloud (6), the cloud adjusts e.g. the zooming ratio (7.1) or the requested ROI. Finally, the video is streamed down to the user device (8) and he/she can watch it (9) making use of all improvements provided by the cloud services like segment- and tag-based browsing.

Figure 6.8 shows the concept of video browsing. On top of the screen three images are seen which represent the different segments of the video. Below them the main video player is displayed. On the right side of the image a list with all available metadata for this video is shown. These segments can now be used to browse the video by clicking on them. The user is then directly forwarded to the corresponding time point. The same happens for the metadata.

The enhancement of mobile video UX in our setup consists of three parts. First, thumbnail cue frames are generated at the transitioning scenes (i.e. events in the video). Figure 6.8 shows the concept of video browsing in more details. The thumbnail seek bar is placed on

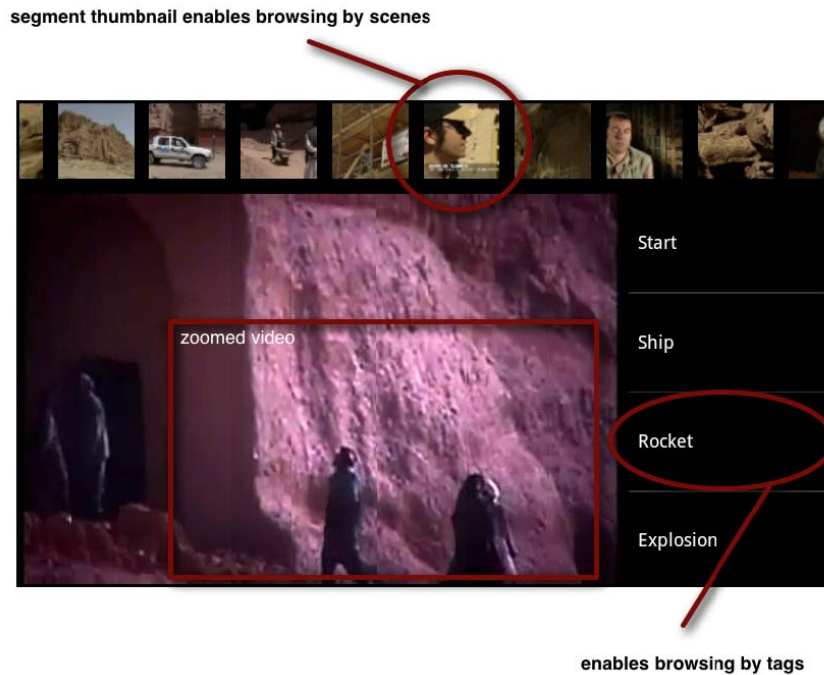


Figure 6.8: Segment and metadata based video stream browsing

the top part. It consists of thumbnails of all the scenes of the video ordered by occurrence. This makes it easy to browse videos. The user can orientate himself by the thumbnails and does not need to wait until the video is loaded at a certain time point. This works well regarding the low bandwidth. As described before, the thumbnails have such a small resolution that they are loaded very fast. Furthermore, a lazy list has been implemented so that it requires even less bandwidth as only currently viewable images are loaded. Clicking on a thumbnail redirects the user directly to the corresponding scene in the video. The user can now search content much faster than in a traditional video player. This again improves the orientation for the user. If the user clicks on an image he is directly redirected to the corresponding time point in the video. Furthermore, the seek bar focuses the current scene and scrolls automatically.

Second, the tag list (right) consists of tags which have been added manually by the user himself, by other users, or generated automatically. Like the thumbnails the tags are ordered by timely occurrence in the video. If a user clicks on a tag, the video stream goes directly to the corresponding position. Both components, i.e. the segment-based seek bar and the tag list are implemented to overcome the mobile UX problem of video browsing on a mobile device.

Finally, the third part of mobile UX improvement contains the video player itself. As device information including screen size is sent to the cloud the necessary zooming ratio can be calculated. Depending on the screen size of the device a zooming ratio is calculated and

depending of the objects a center position of the zooming field is defined. The red rectangle symbolizes the zoomed video. Two persons are recognized by the object recognition service and therefore the zooming field is set at this position. The user just sees this part of the video and can better concentrate on the persons. For future implementations more enhanced classifiers or feature descriptors can be used (e.g. football, etc.) to enable personalized zooming.

Use Cases

As shown on Fig. 6.9, the main and only actor is the mobile user. The user has four possible tasks. The first one is to record a new video with his/her mobile device. He/She will then upload or stream the video to the system/cloud. There it will be processed in parallel using the intelligent video processing services to later improve the user experience. The user can now search or choose a video to watch. After capturing the video he is also able to watch his own video after a short time period. If he/she chooses a video it is now streamed to the mobile device and gets a real-time adaptation by zooming the video depending on the screen size. Sometimes the user wants to browse the video. The browsing process is enhanced by showing the distinctive scenes in a video.

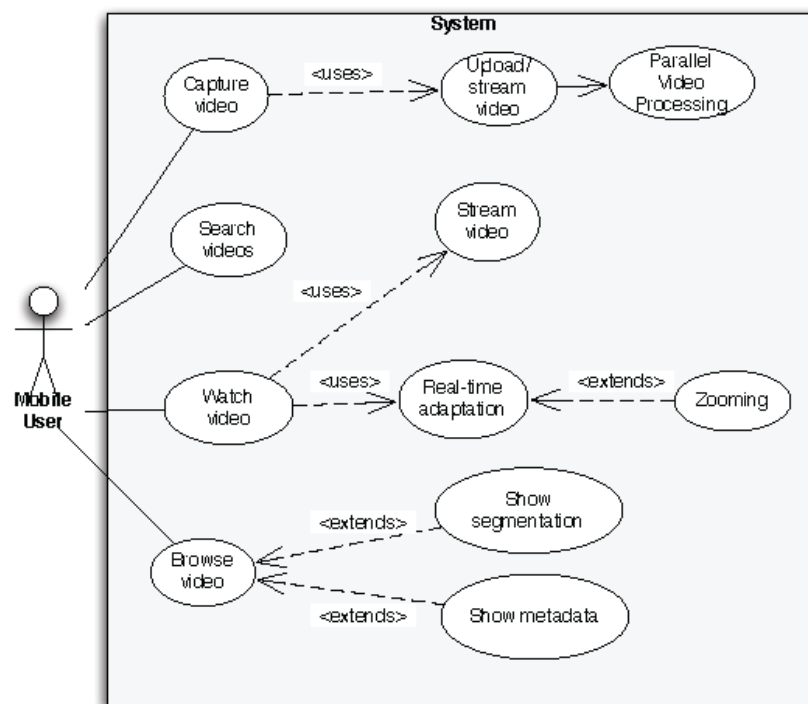


Figure 6.9: MVCS use case diagram

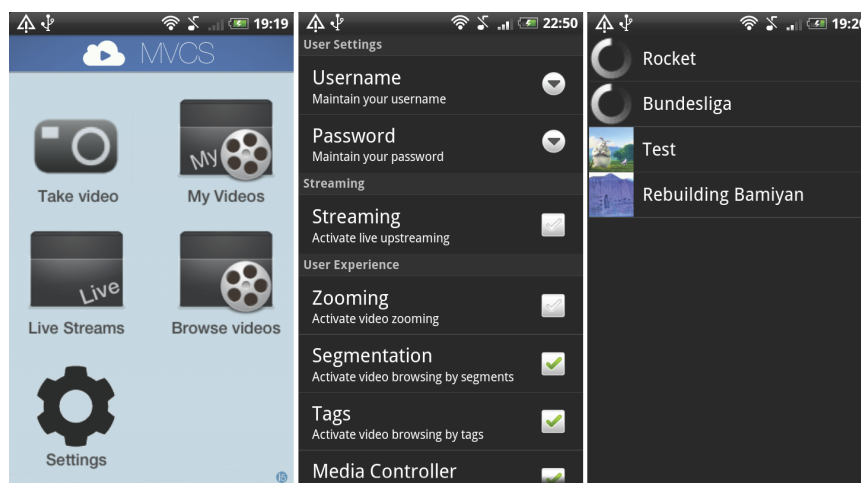


Figure 6.10: User interface of a MVCS-based mobile application

Enhancement of Mobile Video User Experience

The most important part of the mobile client is the video player. Here is where all the tools for improvements of the mobile user experience matter. It displays the segments, tags and also the zoomed video content, as show on Figure 6.8.

Figure 6.10 displays the main user interfaces on the mobile client application. On the leftmost screenshot, main actions are shown. The middle screenshot displays the preferences. This preferences activity provides the user with several setting possibilities. Besides being able to change username and password, the user can also adjust several user experience settings. The user can switch on and off the different video player controls like browsing by segmentation, by tags or the classical seek bar. In addition, automatic zooming and real-time upstreaming while taking a video can be switched on or off. The rightmost screenshot shows the browse activity implemented as lazy load list. The *synchronization handler* has a similar functionality like the metadata handler. Device information, such as screen size, device model and bandwidth, is managed and send via the XMPP connector. This information is then used in the cloud for intelligent zooming and can be used in future for other intelligent video processing services.

On the cloud side, the XMPP service handles all major communication between the client and the cloud. The transcoding service is an interface for the cloud video processing tasks. It uses the FFMpeg library [FFMp13] for transcoding of the video into different formats and to generate thumbnails. The zooming service is responsible for cropping the video. It provides standard zooming functionality like zooming to the middle of the video and more complex zooming functionality based on the object recognition service. The segmentation service recognizes scenes in the video and creates a list of time points of the scenes. Additionally, it utilizes the transcoding service to create the thumbnails for the segments. This is realized by using the FFMpeg. The shotdetect library for scene detection. The object recognition

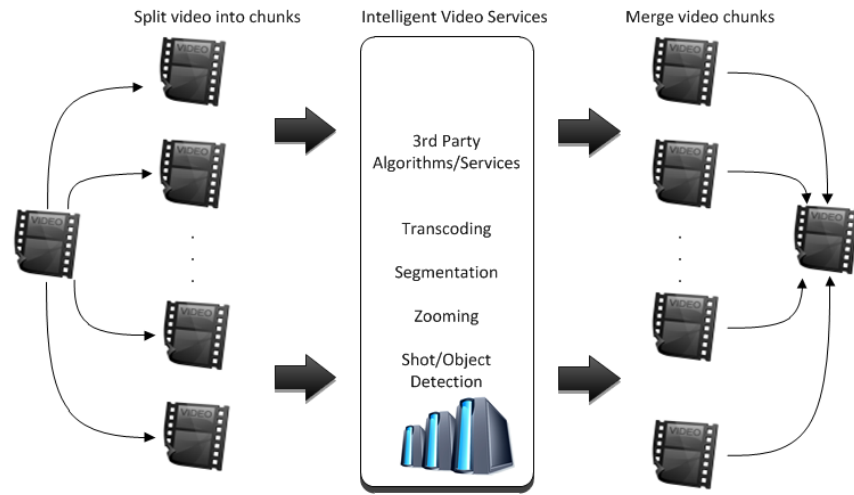


Figure 6.11: Video processing workflow

service is necessary for realizing ROI-enhanced zooming. The service recognizes objects in a video and therefore the center of the zooming region can be adjusted. Recognizable objects are, for example, faces, profile faces, etc. In summary, the video transcoding and zooming services are handled by FFmpeg and the segmentation and object recognition services are mainly handled by OpenCV algorithms [Brad11].

The object recognition is realized with the Java wrapper for OpenCV, i.e. JavaCV. Standard Haar classifiers [LiMa02] are used to recognize faces. The metadata service is responsible for handling metadata. Part of the metadata are tags and technical information like length. This metadata is stored into a relational database (MySQL).

The video service is responsible for the video content storage, and splitting and merging of the video for parallel processing. The parallel processing service is responsible for creating, starting, stopping instances, handling the queue and distributing the processing tasks multiple virtual machine instances.

All steps of the video workflow are basically based on the approach by Pereira et al. [PABE10]. The first step of the video processing workflow is the splitting step. In general, after the upload of a video or an incoming stream, the video is split into several chunks (see Fig. 6.11). The chunks are then processed in parallel by the intelligent video processing services of MVCS. After the processing of the video is finished, the chunks are merged again to a single video. The main problem of splitting a video is the fact that we have to split a video and to merge it again without any loss of synchronization. For instance, taking a look at the H.264 codec, a split in a frame differently than a key-frame (B or P frames) could end up in an useless video file as the other frames depend on information registered in the key frame. Another problem comes with the synchronization of audio and video as the frame size of each one may not be equal.

Evaluation

The evaluation is divided into two parts. First, mobile user experience improvements such as zooming, segment-based and tag-based browsing are evaluated in a user study. Second, the advantages of video processing in the cloud are compared, especially the advantages of chunk based processing in comparison to single file processing.

MVCS client has been implemented for the Android platform. We used HTC Desire smartphone with Android OS version 2.2, 3.7 inch wide screen multi-touch display with 480x800 pixels and 252 ppi pixel density, 1GHz CPU and 576MB RAM.

User Study

Three representative video types were chosen. The first type consisted of sports videos of NFL American football games and German Bundesliga soccer games. The second type consisted of talks from TED conferences [TedV13]. And the third type consisted of documentary films.

We analyzed the effect of zooming, the effect of segment based and tag based browsing on mobile user experience. To compare the different tasks during the user study of user experience of MVCS, the NASA Task Load Index (NASA-TLX) questionnaire [HaSt88] was used. NASA-TLX is usually used to assess the workload on users working with different human-machine systems, in our case the mobile client of MVCS. It is a multi-dimensional rating procedure which derives an overall workload score on the basis of weighted average of ratings on six subscales. The subscales are Mental Demands, Physical Demands, Temporal Demands, Own Performance, Effort and Frustration. Scales like Frustration, Mental Demands are very important for the mobile client of MVCS as searching certain scenes can be mentally demanding and frustrating. We used NASA-TLX to compare the standard video player on Android device with video browsing and zooming enabled player of MVCS.

Twelve persons participated in the evaluation of mobile user experience. The participants were between 23 and 54 years old. They had a mixed smartphone usage background, ranging from individuals who did not possess a smartphone to the ones who watch YouTube videos on their devices everyday.

Each user had to watch each video twice. The first time with zooming enabled and the second time without zooming. The order of playing is changed for every user, in order to minimize the influences of the playing order for the overall results. After each video, the user filled in the NASA TLX Index rating scale after each video playback and weighted the scales after watching all videos. The same procedure was repeated for the segment and tag based browsing. However, the tasks there were extended to find two certain scenes in a video with both the native mobile player and with the MVCS player.

Starting with the analysis of the results of the NASA TLX workload assessment we can prove the improvements by zooming, and segment- and tag-based browsing. The results of the evaluation shown in Figure 6.12 reveal a trend that the workload is lower for the zoomed

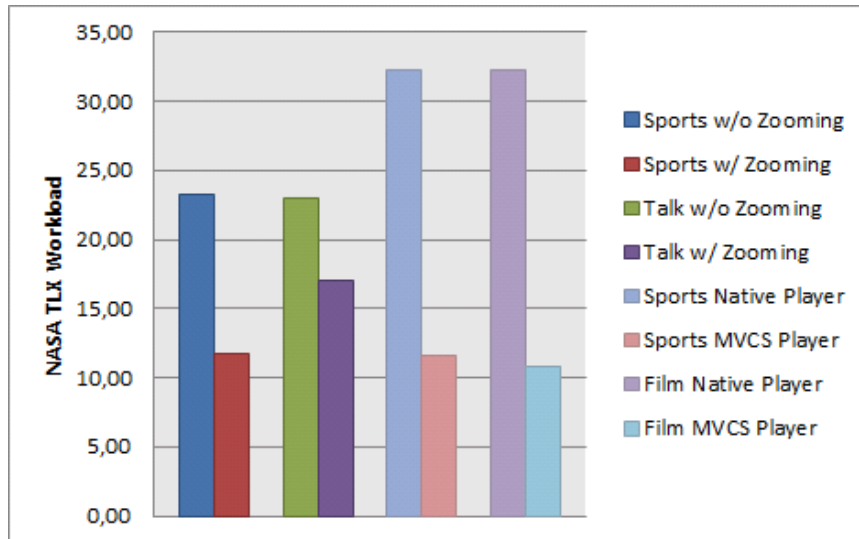


Figure 6.12: NASA TLX user workload

videos. The TLX workload values show that the zooming approach better fits the sports videos. For the talk videos the workload is reduced by 26 % using zooming and for the sport videos the workload is reduced by 49 %. The results lend support that zooming is a good solution to improve mobile user experience for mobile video. The analysis of the video browsing results shows a larger improvement of mobile user experience with much lower workload than the traditional video. Compared to the zooming approach, the type of video content has a big impact on the workload. The workload for the documentary film videos is reduced by 67 % and for the sports video by 64 %.

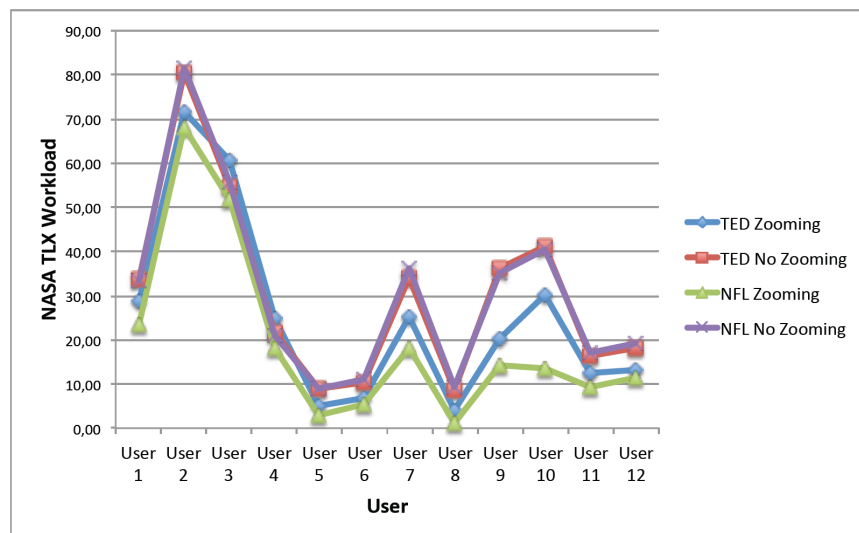


Figure 6.13: NASA TLX user workload at zoomed videos

Beginning with the results of the zooming evaluation, Figure 6.13 shows that the users have different views how to rate the scales. Nevertheless, because of the weighting of the scales, the results are comparable. Thereby, a trend can be identified that the workload is slightly lower for the zoomed videos. Analyzing the different video contents, it looks like the zooming approach better fits the sport video.

Continuing with the analysis of the video browsing results, Figure 6.14 shows an even bigger improvement of mobile user experience. The lines for both videos using the MVCS video player show a much lower workload than the traditional video. In comparison to the zooming approach the video content does not seem to have such an big impact on the workload. Remembering the task of the user to find certain scenes in the video, it can be concluded that mobile user experience is improved for the second scenario.

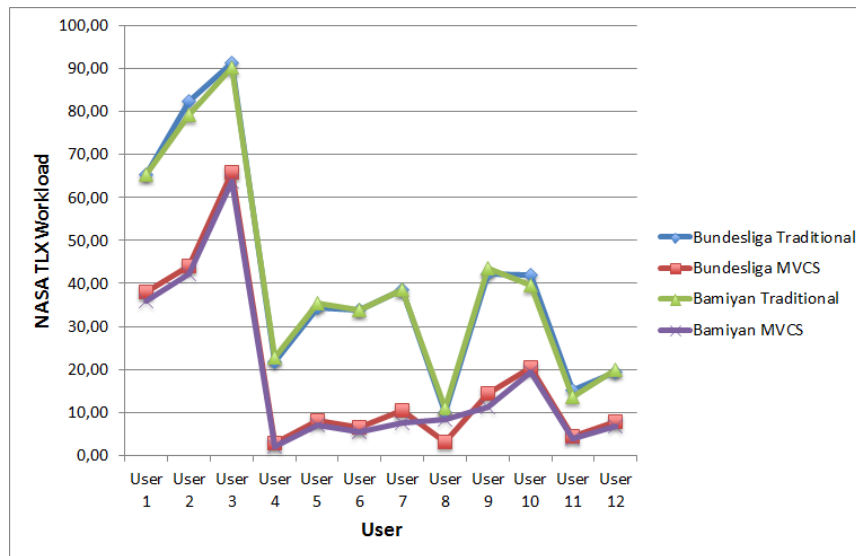


Figure 6.14: NASA TLX user workload at video browsing

Cloud Evaluation

One of the main advantages of the cloud infrastructure for the intelligent video processing services is the accelerated processing/adaptation time. To evaluate the MVCS cloud part three videos with different lengths were chosen. The first video is processed as one chunk on one instance, the second one as three chunks on three instances and finally the third one as five chunks on five instances. Amazon EC2 small instances were used as cloud infrastructure. For every task the processing time is measured. For the chunk-based approach this means the process from splitting the video into chunks, processing them and merging them again. The MVCS cloud evaluation proves that splitting the video into chunks instead of processing it as a single file is a good solution to deliver faster video processing for near real-time delivery.

Figure 6.15 presents a comparison of all cloud evaluation results. Here again, we see that the processing time for the single file approach is increasing with the length of the video. For

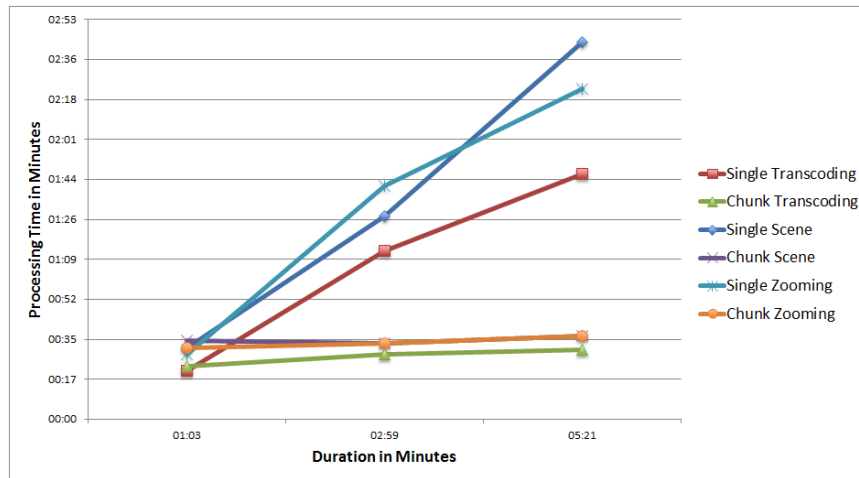


Figure 6.15: Comparison of cloud processing time

the chunk-based approach we see that the processing time is increasing very slightly. The small increase can be explained that merging the chunks to one file takes a little more time for every additional chunk. Nevertheless, our system has also limitation. The user study showed that zooming has to be improved as some users lost the context of the video due to the zooming. The number of evaluation participants and types was limited. A real-world evaluation over a longer period is needed to assess the perceptual quality of the delivered video streams. The MVCS cloud evaluation proves that splitting the video into chunks instead of processing it as a single file is a good solution to deliver very fast and intelligent video processing for nearly real-time delivery.

Discussion

Combining both results of the mobile UX evaluation and the cloud evaluation it can be said, that offloading the CPU-intensive processing to the cloud is a good way to improve mobile user experience. Already a video of about 5 and a half minutes requires processing of about 7 minutes on a small Amazon Web Services EC2 instance [AEC2]. A small EC2 instance can be compared to an average desktop computer or notebook and has much more processing power than a mobile phone. Hence, doing all this processing of the video services would take even more time on a mobile phone. This is far away from delivering near-real-time improvements. Keeping in mind that a cloud infrastructure provides theoretically unlimited processing power, storage and number of instances, the processing time can be kept at about 30 to 40 seconds for a video on a small instance. Using a cloud infrastructure with more powerful instances and several CPU cores per instance could even lower the overall processing time to a minimum resulting in a theoretically real-time mobile UX improvement. If the user is additionally live streaming the content to the cloud instead of uploading it MVCS the overall process is not so dependent on the bandwidth and does not have to wait until the upload it finished.

Furthermore, the evaluation of zooming and segment- and tag-based browsing shows that both approaches do improve mobile user experience. Especially, the segment and tag-based browsing was received well by the test users. As the zooming approach was received a little be more mixed, it requires improvements. It can be observed that especially the context of the video is a problem for video zooming and that a more personalized or dynamic zooming is wished.

In summary, combining both results of the mobile UX evaluation and the cloud evaluation, offloading the CPU-intensive processing to the cloud is a reasonable way to improve mobile user experience.

6.1.3 MACS: Adaptive Computation Offloading into the Cloud

MACS stands for a collection of mobile augmentation cloud services that can act as a middleware for adaptive computation offloading. The architecture for mobile offloading is described in Sec. 5.3.2. We evaluated our MACS framework with two use case phone applications. The first application implements the well-known N-Queens problem. It is chosen because the performance bottleneck represents a pure computation problem. This use case can easily show the overhead introduced by MACS middleware. The second application involves face detection and recognition in video files. This use case involves lots of computation, but also requires much more memory resources to process and obtain results.

The second case can process a video file, and detect faces from the video file, cluster them and provide the time point cues for video navigation. The results can then be used for faster video navigation on small screen devices (Figure 6.16). The video file is processed with OpenCV ² and FFmpeg ³ libraries. In the processing, faces in the video file are detected by the existed implementation in OpenCV, and then the detected faces are recognized by the method proposed by Turk and Pentland [TuPe91], and after that, the faces are clustered.

In the implementation we used JavaCV ⁴ for video processing. When the application gets the results from the processing, it shows all detected faces as a clustered view. The user can select a cluster, and then navigate to the time points where that face occurs in the video. Thus, the application can accelerate navigation in a video based on persons that occur within.

The hardware we are using in the evaluation is as follows. A Motorola Milestone mobile phone based on Android platform 2.2 is used in the evaluation. A desktop computer which includes quad-core CPU acts as a cloud provider that can host the offloaded computation. The details about the hardware components for the mobile device and desktop computer are shown in Table 6.2.

²<http://opencv.org>

³<http://ffmpeg.org>

⁴<http://code.google.com/p/javacv>

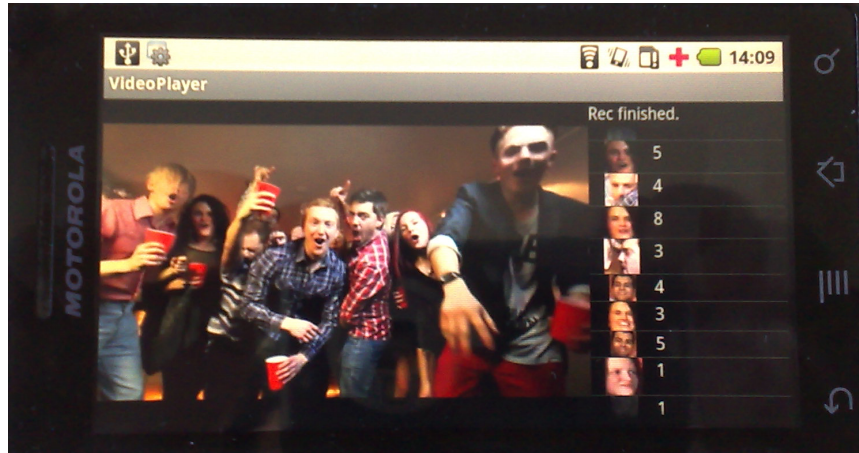


Figure 6.16: Snapshot of a MACS prototype application for face detection and recognition

Table 6.2: Hardware components of mobile device and cloud node

Hardware Component	Milestone	Cloud Node
Processor	ARM A8 600 MHz	Quad-Core 2.83GHz
Memory	256MB	8GB
WLAN	Wi-Fi 802.11 b/g	N/A
OS	Android 2.2	Windows XP x64

Network Topology

While offloading services to the remote cloud, the mobile phone connects to a nearby access point. Since the wireless local area network is encrypted with Wi-Fi Protected Access 2 security protocol, the data speed is not as fast as non-encrypted considering of the overhead introduced by the security protocol. The desktop computer is connected to the Internet directly by network cable, whose bandwidth is 100 Mbps.

Energy Estimation Model

We adopt a method as the one proposed by Zhang et al. [ZTQ*10], a power model for an Android phone and a measurement application for the energy consumption on the Android-based mobile device on the fly. Using their software, the energy consumption of each hardware component of the Motorola Milestone such as LCD, CPU and Wi-Fi can be measured separately (see Table 6.3).

Results of the Use Case 1

The algorithm we use is by Sedgewick and Wayne⁵. The basic idea is to use recursion and back-tracing to enumerate all possible solutions. Although it is not the best algorithm, it is

⁵<http://introcs.cs.princeton.edu/java/23recursion/Queens.java.html>

Table 6.3: Estimated energy consumption of mobile device

Hardware Component	Estimated Energy Consumption (W)
Processor	0.4 (Idle: 0.05)
Wi-Fi	0.75 (Low: 0.03)
LCD	0.9

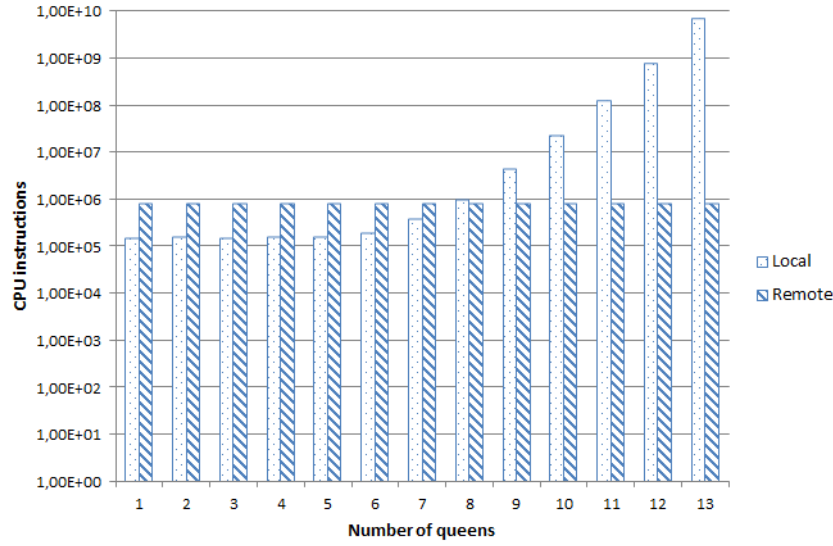


Figure 6.17: CPU executed instructions of N-queens

often used for solving the N-queens puzzle. It is clear that with the increase of N , much more steps are spent to find solutions, which is extremely time-consuming for the mobile device.

We run the N-Queens on the local device and offload to the remote cloud separately, for $N = 1$ to $N = 13$. As when $N = 14$, it will take hours to finish on the local device, it is not realistic not to be offloaded while doing computation after $N = 14$. Figure 6.17 shows that how many CPU instructions are executed while calling the specific calculation service. For local execution, the instructions number increases with the raise of the queens' number, especially after $N = 8$. The reason is that the applied algorithm contains recursion, with the increase of N , much more search steps are computed. But the instructions number of remote execution is always stable at the same level, which indicates that most of the instructions are offloaded to the remote cloud and relief the load from the local device.

Figure 6.18 shows the time duration of execution of the specific calculation service. From $N = 1$ to $N = 9$, the execution speed on the local device is acceptable compared with the remote speed and to run the method locally is better, but after $N = 10$, the remote speed dominates to be the better option as the computation time dominates the total time in the rest cases, and the remote execution speed is also relative stable, there is no huge variation

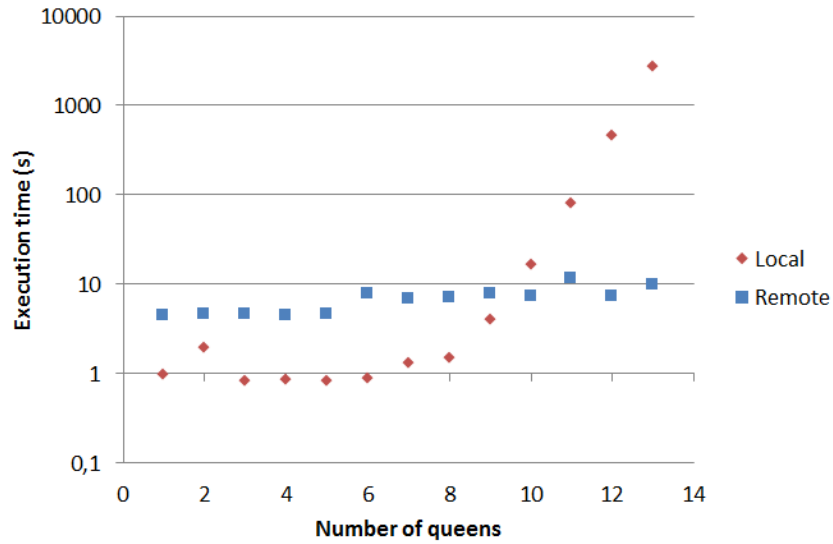


Figure 6.18: Execution time of N-queens

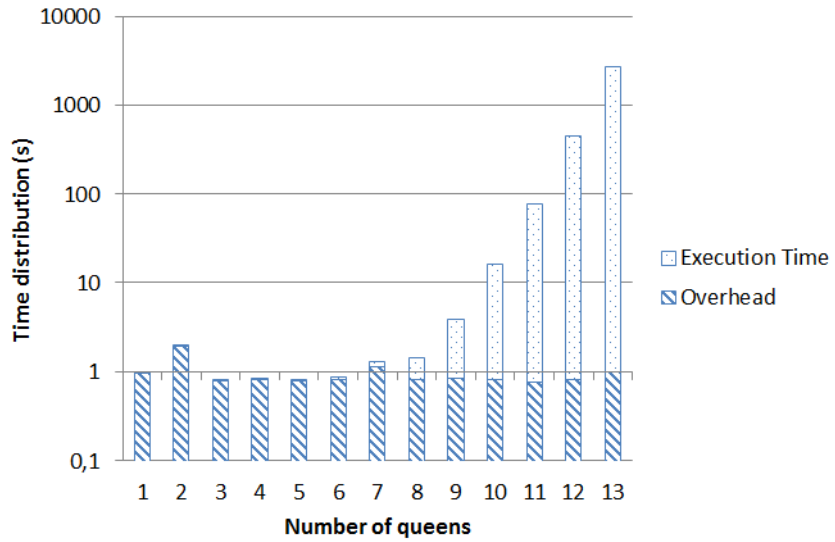
for remote speed.

Figure 6.19 shows the different time parts, which are made up of the total spent time. With the increase of the queens number, the local execution time increases outstandingly, especially from $N = 9$, the execution time of calculating solution occupies more than half of the total time. Meanwhile, the overhead, our framework brings, stays constant. As for the remote execution, the overhead is broken down to three parts, one is the package offloading time, one is the decision maker time, the rest one is the residual overhead. It shows that our decision model costs only little time to finish the determination, and the transmission time of remote package occupies also few parts of total time, since the remote package is small. The execution time of solving the N-queens is relative stable, except for the $N = 11$, which is a deviation during the execution and measurement.

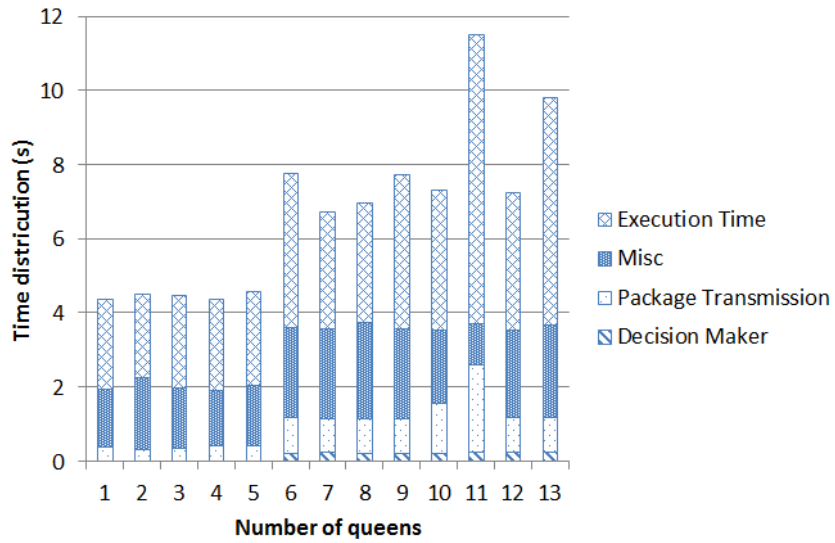
The Figure 6.20 shows the results of consumed energy with and without offloading. As for the local execution, most of the time is spent on computation, since our energy model involves CPU and LCD, and the LCD is always on while computation, so that the energy consumption of CPU and LCD dominates the total energy consumption of local execution. The execution time is significantly increased from $N = 9$ compared to the remote execution, which leads to the highest energy value. In contrast to those, the remote execution time is nearly stable, so that the consumed energy is almost at the same level.

Results of the Use Case 2

Six video files are used in the evaluation. All of them belong to a same original video file with different length of time, 10, 20, 30, 40, 50 and 60 seconds (see Table 6.4). The video



(a)



(b)

Figure 6.19: Total time distribution of N-queens: (a) local execution and (b) remote execution

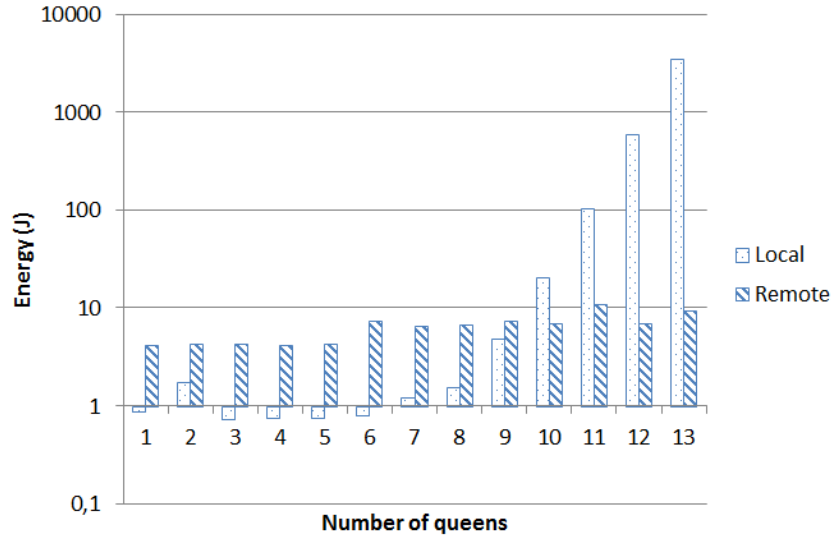


Figure 6.20: Energy consumption of N-queens

Table 6.4: Video duration and file size

Duration (seconds)	File size (bytes)
10	1864984
20	3864612
30	5827420
40	7754219
50	9633240
60	11584020

resolution is $720 \text{ pixels} \times 480 \text{ pixels}$, the fps is set to 30, the overall bit rate is 1500 Kbps and the video is compressed with MPEG-4 format, 3GPP Media Release 5 profile. The audio codec is AAC, and the bit rate for audio is 30 Kbps.

In order to get an more accurate estimation of execution time which is used in the model, we first run the face detection services locally, and keep track of the spent time (second) and the file size (bytes), and then a linear regression is used to reflect the relationship between the spent time and the file size. Considering the CPU count provided by Android API, it can only be used to make estimation on the execution which involves no native calls, we don't directly use that count, but focus on the execution time. The regression shows that,

$$Time = 0.0005 * FileSize - 246.09 \quad (6.1)$$

We use this heuristic equation in our model to make determination of the execution time.

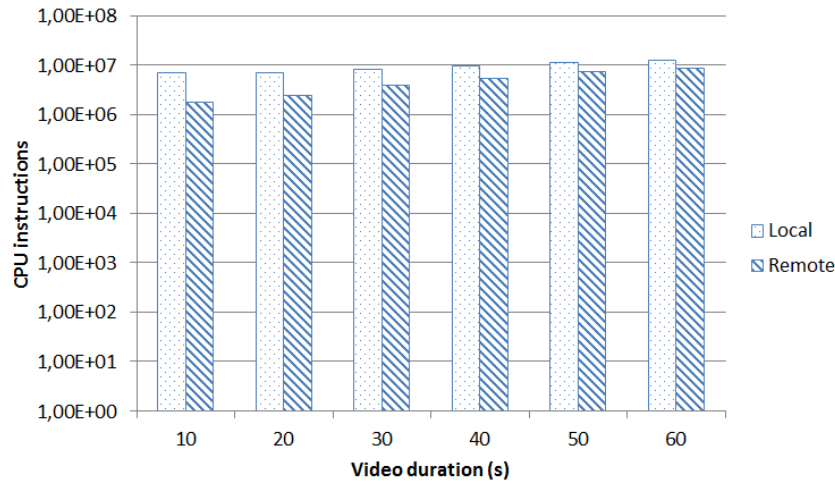


Figure 6.21: CPU executed instructions of face detection

Table 6.5: Video duration and speedup of face detection in video file

Duration (seconds)	10	20	30	40	50	60
Speed up	×20	×26	×23	×28	×28	×29

The comparison of the number of local and remote executed CPU instructions is shown in Figure 6.21). However, one must note that the Android API does not record the instructions which are executed by native methods, so there is only slight difference between local and remote execution.

It can be seen clearly that the execution time is reduced with offloading compared with local execution only (see Figure 6.22). Even dealing with a ten-seconds video file, the local device spends more than 15 minutes on processing and detecting, but the corresponding remote offloading takes less than a minute. Each time the computation is offloaded to the remote cloud, the execution speed can be reduced to more than 20 times (see Table 6.5). It is absolutely not acceptable to let the CPU of local mobile device 100% load for such long time, and it confirms that the video processing task is still a huge burden for the mobile device.

With the big difference between local and remote execution time, it is apparent that the local energy consumption is worse than the remote ones, because most of the time are spent on CPU and LCD, which are the top two of energy consuming components (see Figure 6.23). Table 6.6 depicts the energy savings with offloading, i.e. more than 94 percents of the energy can be saved due to offloading.

Figure 6.24 describes the composition of the local and remote total spent time in details. As the execution time increases with the bigger video file size, the overhead our framework brings occupies about only 0.1%, which can be nearly omitted. Regarding to the remote

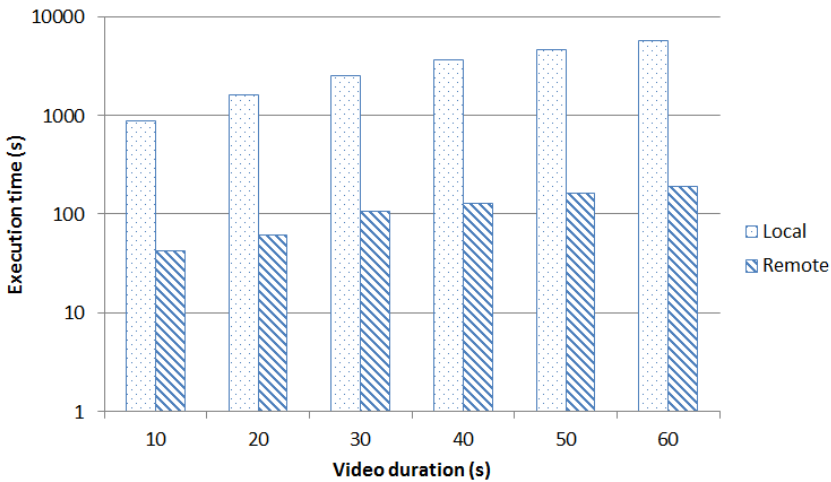


Figure 6.22: Execution time of face detection

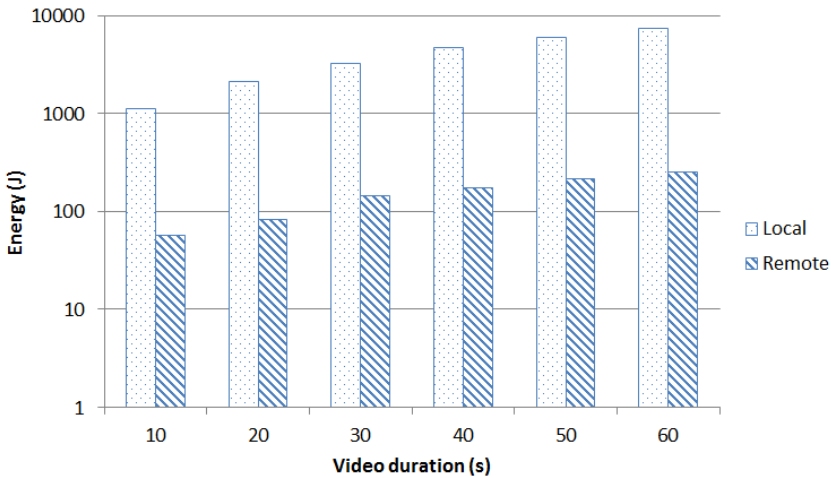


Figure 6.23: Energy consumption of face detection

Table 6.6: Video duration and energy savings

Duration (seconds)	10	20	30	40	50	60
Energy save (%)	94.98	96.07	95.59	96.37	96.33	96.55

execution, the total spent time consists of execution time, needed file transmission time, package transmission (service offloading) time, decision maker time and so on. With the increase of the video file size, the file transmission time also raises, but compared to the total time, it is not significant. The decision maker does its determination in less than 1 second, which is only 1 percentage of the total spent time. The total overhead our framework brings is about 5 percentage of the total time, which is acceptable considering about the speed up and energy save above.

The face clustering can only be done on the remote cloud because of the software limitation on the local mobile device. Most of the execution time is spent on building/rebuilding training set. If the training set is already available before the remote execution, then the estimated execution time can be significantly reduced.

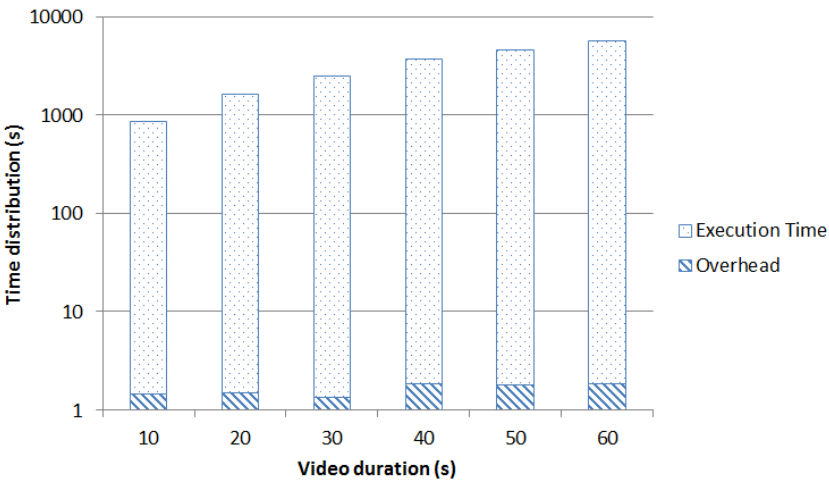
Discussion

Offloading perhaps is not the suited for every mobile applications, but from the results of the two use cases, we see that when an application uses complex or time-consuming algorithms such as recursion, by offloading those parts into the cloud, time and energy consumption are reduced, so that the local execution time is reduced to an acceptable level. Offloading can lower the CPU load on a mobile device significantly. It can also save lots of energy, which indicates that the battery time can be increased compared to the local execution, as shown in the second use case, where more than 90% of energy is saved and the calculation speed is up to 20 times over local execution.

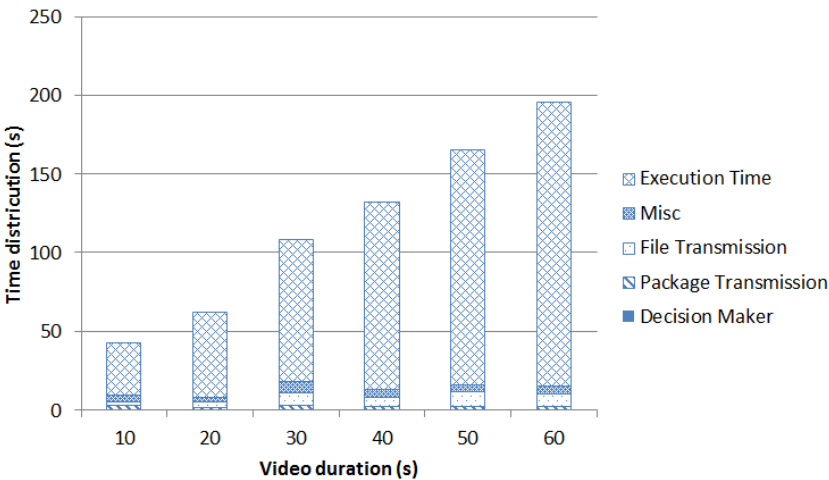
The results also prove that the overhead of our framework is small and acceptable with the increase of needed computation, it is better to push those computations which cost considerable resources to the remote cloud. But for the small N in the N -Queens problem, the overhead occupies almost half of the total execution time because of the needed computation is small so that it takes only little time to obtain the results. This shows a clear advantage of local execution over remote offloading when the needed computation is not much. In a word, the more computation is needed, offloading has more advantage. However, since we use Wi-Fi in the evaluation, the time of sending files and receiving results has small proportion, but if 3G or GPRS are used, the offloading time will surely increase.

6.2 Mobile Multimedia

This section describes advanced services and information systems covering the mobile multimedia services. First, an integration of a mobile and a Web application for ubiquitous multimedia is presented. Diverse media operations are enabled on both mobile and Web clients, while i5Cloud plays the role of a connecting platform. Second, collaborative services between heterogeneous mobile and Web clients are demonstrated. These services



(a)



(b)

Figure 6.24: Total time distribution of face detection: (a) local execution and (b) remote execution

use multimedia metadata as central collaboration artifact. They have been evaluated in a digital documentation scenario in cultural heritage management.

6.2.1 AnViAnno: Ubiquitous Multimedia Management

AnViAnno is an mobile application for context-aware acquisition, sharing and consumption of mobile videos and images (see Fig. 6.25). Additionally, AnViAnno captures the device (spatio-temporal) context which is further used to support the semantic annotation on mobile devices and web clients. AnViAnno seamlessly integrates with the web-based SeViAnno, in regards to the multimedia content and metadata. Actually, AnViAnno can be considered as the mobile counterpart of SeViAnno [CRJ*10], an interactive Web platform for MPEG-7 based semantic video annotation. SeViAnno features a well-balanced trade-off between a simple user interface and video semantization complexity (see Fig. 3.9). It allows video annotation based on the MPEG-7 standard with integrated various tagging approaches on multi-granular and community levels.

Figure 6.25 contains snapshots of AnViAnno user interface and functionalities used for context-aware semantic multimedia annotation. Videos recorded with the phone camera can be previewed and initially described. Semantic annotations are realized as MPEG-7 Semantic Basetypes including *Agent*, *Concept*, *Event*, *Object*, *Place* and *Time*.

The user generated annotations are further used to navigate within video content or improve the retrieval from multimedia collections. For example, users can navigate through the video(s) using a seekbar or semantic annotations. The videos and their annotations are exposed to other internal LAS MPEG-7 services [CKJa10] and external clients.

The mobile devices enable users to make the initial metadata enrichment on site during the multimedia acquisition and capture context cues. However, they have limitations in regard to input modes and screen sizes. Our multimedia services build using the i5Cloud enable extension of the metadata management on desktop or laptop computers using the SeViAnno web application. Figure 3.9 depicts a screen snapshot of SeViAnno's user interface featuring a video player (top left), video information and video list (bottom left), user created annotations (top right), and Google map mashup for place annotations (bottom right).

6.2.2 XMMC: Collaborative Metadata Services in Cultural Heritage

In order to evaluate i5Cloud and its services from the user and community facet, we apply a use case scenario with regard to professional community's requirements. The prototypical implementation and testing of XMMC components were supported by a master thesis [Aksa11].



Figure 6.25: Screen snapshots of AnViAnno

Use Case Scenario

A simulated research team consisted of experts on different fields such as archeology, architecture, history, etc., which are documenting an archaeological site. Team members were distributed on the field. Figure 6.26 outlines the scenario. First, a documentation expert discovers some artifacts on site and documents them with photos and videos. He also tags the multimedia content with basic metadata like name and description. The multimedia is stored to the *Collaborative Multimedia Cloud*, i.e. using the mobile multimedia services running on top of i5Cloud. Other experts, on-site or remote, join the session to collaboratively annotate the multimedia in real-time for different aspects. Therefore, a collaboration session is established by the Collaborative Multimedia Services. Then, for example, an architecture expert annotates the origin date/period of an artifact. This annotation is propagated to all team members immediately. Then, a historian augments that annotation, since he has deeper knowledge in these scopes. His correction is also pushed to all others seamlessly.

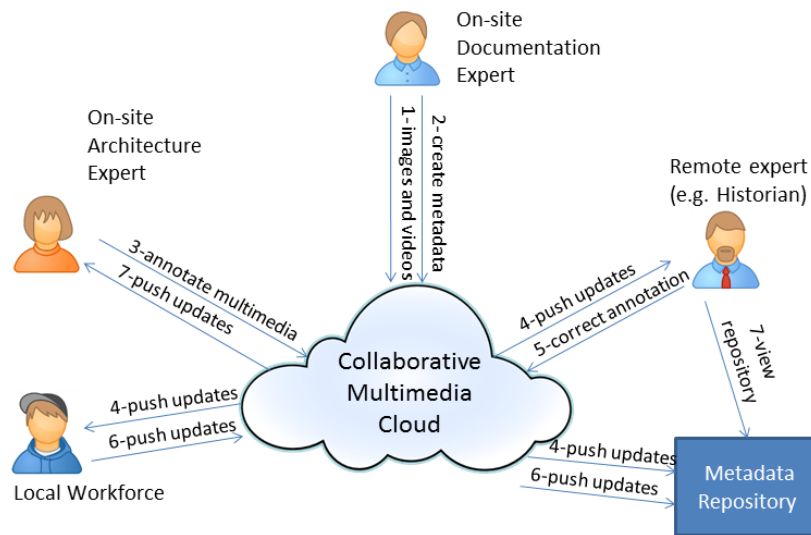


Figure 6.26: XMPP-based mobile multimedia collaboration

The mobile multimedia cloud platform comes with real-time support, multimedia acquisition and sharing with other users, collaborative multimedia metadata annotation and integration with existing metadata repositories.

Real-time Collaboration

We use XMPP as a main communication protocol to support real-time collaboration. Our XMPP-based mobile multimedia collaboration (XmmC) services run on top of i5Cloud.

Multiple XmmC clients can bi-directionally stream XML stanzas over the XmmC i5Cloud services using XMPP channels. The mobile client application also operates as a tool for

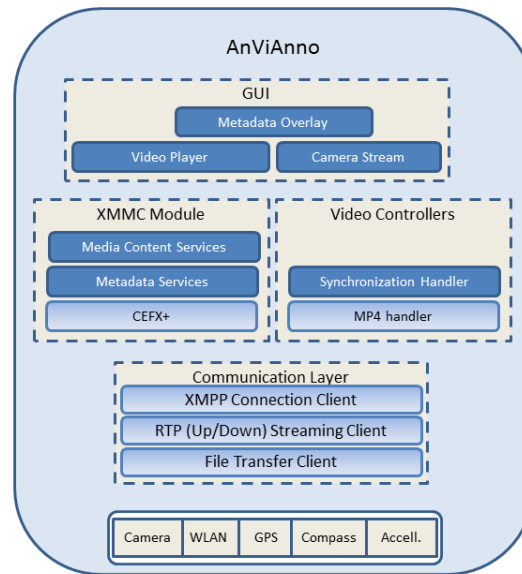


Figure 6.27: Main components of a mobile client for collaborative multimedia processes

multimedia acquisition, annotation with metadata, and multimedia content and metadata consumption (see Fig. 6.27).

On the cloud side, the XMPP communication is conducted via an Openfire XMPP Server [Real11b] (see Fig. 5.1). The system is build on top of the XMPP server as an Openfire plugin. Two XMPP modules are responsible for the XMPP communication between clients.

First, Media Catalog Module is responsible to create, retrieve, update and delete operations for multimedia content and metadata. The Media Catalog Module persists the basic multimedia-related data on a relational database and ensures consistency of this data.

Second, Collaborative Metadata Module handles metadata related services, i.e. metadata management and synchronization of the annotation metadata in real-time among all collaborating client applications. The concurrent XML editing service is based on the Collaborative Editing Framework for XML (CEFX+) [Gerl07, Voig09]. CEFX+ enables concurrent editing of XML documents in real-time using operational transformation algorithms [SuEl98]. The synchronization is done by keeping a copy of the XML at every client in the session and then in case of edit operations the service ensures timely updates on the copies. If some conflicts exist, it resolves them and broadcasts the changes with a message for conflict resolution on all clients.

During the synchronization of all working document copies, a copy is also situated at LAS-MPEG7 Integration Service so that the service would act as a client that synchronizes the XML metadata file and calls the related LAS MPEG-7 Semantic Base Type Services whenever an XML document is updated. Thus, XmmC achieves interoperability with pre-existing and standard-compliant multimedia repositories (such as the one described in [CJK*09]).

Figure 6.28 demonstrates the sequence diagram of collaborative multimedia annotation. First, Mobile Client-1 sends a `<collanno-req>` get IQ in order to retrieve metadata XML of a multimedia. Then, i5Cloud Collaborative Annotation Module (CAM) gets the corresponding XML file from Collaborative Editing Service (CES) and sends back the file via XEP-096 SI File Transfer extension of XMPP [MMES04]. Once the transfer of the file succeeds and is acknowledged from client side, CAM sends back `<collanno-req>` result IQ so that clients can start annotation.

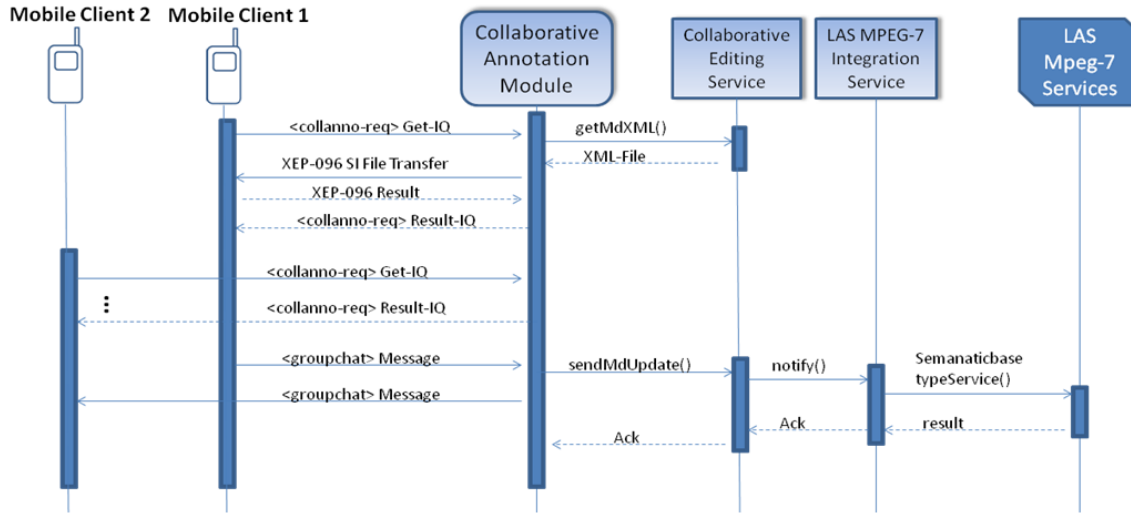


Figure 6.28: Sequence diagram of collaborative annotation

Meanwhile, Mobile Client-2 also wants to participate into the collaborative annotation session. It sends a `<collanno-req>` message and is followed by the same procedure that happens for Mobile Client-1. Both clients share the editing session. Client-1 performs an insert, delete or update operation. The operation is propagated to all participants of the editing session via `<groupchat>` message of the XEP-0045 Multi-User Chat extension of XMPP [Sain08]. The XML document at i5Cloud is also updated. Finally, CES notifies the LAS MPEG-7 Integration Service for the changes done.

Annotation Operations. Operation message stanzas represent the information that is needed to be provided for the CMAX algorithm. In the following the abbreviations that are used at operation messages and their corresponding semantics are explained.

- *p*: position identifier of the parent node at XML Document which is defined at *cefx:uid* attribute
- *ci*: client ID of the operation sender
- *sv*: state vector at the client which is originated before operation is locally executed
- *ba*: before or after, 0 for before and 1 for after

```
<message id="8Dpbh-26"
to="xmmc@merian.informatik.rwth-aachen.de/Smack"
from="3b90aa319d44@conference.merian...de/test_1"
type="groupchat">
<body>
  <basetype type="time" cefx:uid="101">
    <name cefx:uid="201">
      Second World War</name>
    <date cefx:uid="301">
      1997-09-24T00:00:00:000F1000+01:00
    </date>
  </basetype>
</body>
<x xmlns="jabber:x:cefx#ins"
  p="100" ci="1" sv="0,1"
  ba="1" fn="null"/>
</message>
```

Listing 6.1: Insert operation message stanza

```
<message id="8Dpbh-27"
to="xmmc@merian.informatik.rwth-aachen.de/Smack"
from="3b90aa319d44@conference.merian...de/test_1"
type="groupchat">
<body />
<x xmlns="jabber:x:cefx#us" p="201" ci="1" sv="0,2"
  ba="1" nt="txt" text="Second World War" fn="null"/>
</message>
```

Listing 6.2: Update operation message stanza

- *fn*: position identifier of the fix node at XML Document which is defined at *cefx:uid* attribute
- *nt*: node type possible values *txt* for text node or *attr* for attribute
- *ins*: insert operation
- *del*: delete operation
- *us*: update state operation

There are two main XML nodes *body* and *x*. Body element as seen at Listing 6.1 carries string representation of a new XML node with sub attributes and nodes. On the other hand, *x* node consists of operation specific attributes. It is crucial to have minimum payload for various operations in order to fulfill minimum battery consuming behavior. Therefore, only required information is transmitted with the stanzas. For instance as shown at Listing 6.2, the payload of the stanza consists of only atomic modified node instead of sending modification events with the whole corresponding semantic annotation.

Collaborative Mobile Augmented Reality

Mobile augmented reality becomes increasingly feasible on inexpensive hardware at mass-market effect. Augmented Reality (AR) is a natural complement to mobile computing, since smart phones can change their user interface so that the physical world becomes a part of the user interfaces itself. Accessing and understanding information related to the real world becomes easier. This has led to a widespread commercial adoption of MAR in many domains like gaming, cultural heritage, assisted directions, marketing, shopping, education and instruction.

Furthermore, in order to support diverse digital content several popular MAR applications like Layers and Wikitude have shifted from special-purpose applications into MAR browsers which can display third party content. Third party content providers use predefined APIs which can be used to feed content to the MAR browser based on context parameters. Hence, they are relieved from the technical burden of managing AR applications.

However, such MAR browsers have limited support for mobile real-time collaboration (MRTC), i.e. to enable users to collaborate and communicate with each other in real-time. Most MAR browsers operate on common service oriented approaches like REST, SOAP and HTTP which are inadequate for real-time collaboration. Moreover, these approaches provide only pull functionality, i.e. the request for data transmission is actively initiated by the client without a genuine two-way peer-to-peer communication. Moreover, the semantics of digital objects (i.e. multimedia) in MAR is barely considered beyond the location parameters. A MAR system empowered with collaborative actions around common data can increase the user experience in many professional domains enormously.

Augmented reality can be briefly defined as augmenting the physical life by computer based generated 2D/3D graphics or sound. It has been significantly evolved in recent years. The classical human computer interaction experience has been step up by providing a combination of digital and physical world perception.

Höller and Feiner [HoFe03] give a detailed introduction to mobile AR (MAR) systems and review some important MAR system considerations. Laya[r][Wang11] is a mobile outdoor AR platform for discovering surrounding of the user. The platform displays the physical world by augmenting it with the digital retrieved information called “layers” via a mobile device. The platform not only provides a location based augmented reality solution, but also an API for third party developers that helps them to create, maintain and publish “layers” that extend the application for their own purposes [Wang11]. However, the major drawback of Laya[r] is the flow of information which is only one way. The client can not push other data beside the location and some parameters which conflicts with collaborative applications. You et al. [YBSe10] tried to overcome such limitations of pull-based MAR browsers in their Mixed Reality Web Service platform. Their platform provides a RESTful interface for 3rd party geo-spatially oriented content. However, their workaround solution of a publish/subscribe system with periodical polling suffers from the same limitations. In contrast, our XmmC framework is build around XMPP protocol extensions, thus ensuring

simple and effective bi-directional XML-based communication.

Augmented reality and basic multimedia metadata, semantic annotations are stored as XML document for collaborative editing. Our collaborative AR metadata editing service uses the consistency maintenance algorithm for XML (CMAX) [Gerl07] algorithm. The synchronization is done by keeping a copy of the XML at every client in the session and then in case of edit operations the service ensures timely updates on the copies. The updates are performed by sending the changes with a message “propagateLocalOperation” to the server which will apply the changes to the server copy. If some conflicts exist, the server resolves them and broadcasts the changes with a message to be made to have consistent versions of the document on all clients.

Figure 6.29 demonstrates the data model which is persisted by XmmC.

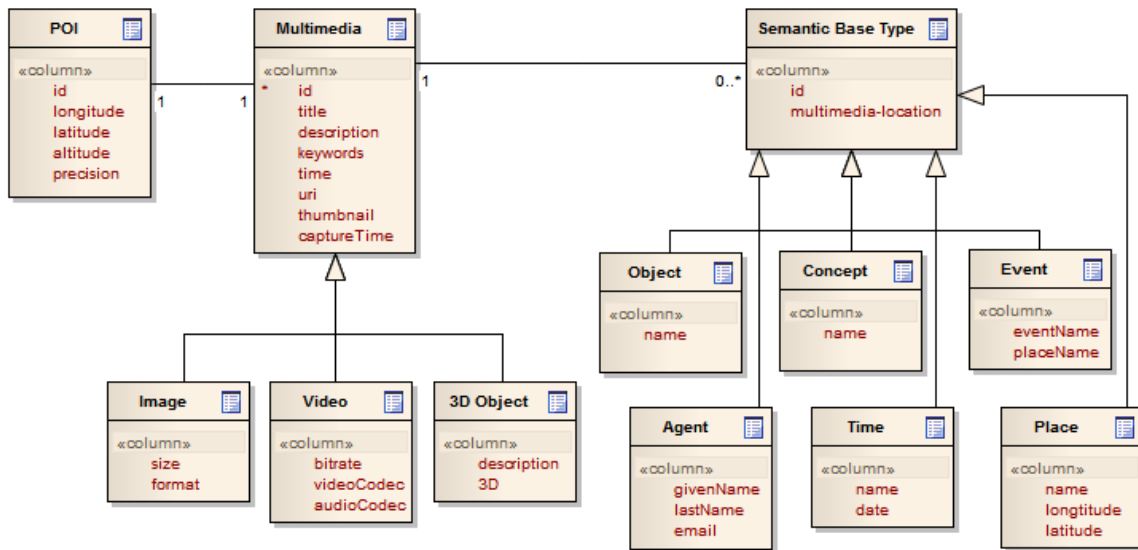


Figure 6.29: XmmC ER diagram of multimedia metadata and multimedia semantics expressed as MPEG-7 Semantic Descriptors

The system uses data type POI for AR purposes. Therefore, users can query the POIs according to the geographical location. The altitude field is used for determining where POIs are displaying on the augmented reality. Every POI has a reference to a multimedia.

A multimedia artifact can be any kind of multimedia data, which can be rendered by the mobile client, e.g. video, images and 3D objects. Title, description and keywords form the basic metadata about the multimedia whereas URI specifies where the multimedia is available and thumbnail keeps the reference to the thumbnail of the multimedia.

Every multimedia artifact can have multiple semantic base types which are used for annotation. Integration with our existing MPEG-7 services [SKJR06] is one of the goals of this work. The annotation model is defined according to the semantic base type def-

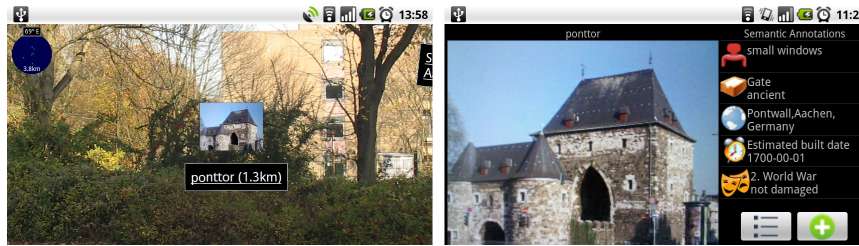


Figure 6.30: XmmC MAR browser

inition used at MPEG-7. This integration provides access to large repository of already semantically-enriched multimedia.

Figure 6.30 shows screenshots of the XmmC mobile AR browser. On the left is the camera preview augmented with nearby POI and their related multimedia artifacts. On the right is the interface with MPEG-7 semantic annotations which can be created, edited and deleted with other mobile clients collaboratively.

We may classify the annotations made by the user, according to what it refers to. More precisely, it can refer to the real object that is documented or the virtual representation of a real physical object which is taken out of its context. For instance, if we talk about creation date of the artifact which can be expressed with time base type than it belongs to the real object. On the other hand, the coordinates of the media at acquisition which is part of the POI data refers to a virtual object. The information about the virtual object is mostly used for technical purposes, like in our case, the geographical coordinates are needed for augmented reality and providing proximity information.

Browsing surroundings augmented with the multimedia is among the main use cases. The AR Browser displays the POIs of multimedia as an overlay on the camera view. Users can navigate to Multimedia Content Activity, by touching a POI. Moreover, it lets the users to filter POIs by changing the radius. The AR Service retrieves a multimedia list from Catalog Service, then filters the list by comparing the calculated distance between mobile client location and acquired multimedia location with user defined radius accuracy and finally returns back to AR XMPP module the list of POIs which are transformed from multimedia that relies inside the defined distance range. The accuracy values of multimedia are considered for location distance based filtering because of inaccurate GPS (Global Positioning System) readings.

Figure 6.31 demonstrates the mobile application user interfaces used in as a collaborative mobile augmented reality tool. XMPP core standard can easily be extended with custom IQ stanzas to enable the communication between client and cloud in this specific domain.

In order to offer support for managing the multimedia content in Web browsers, we extended XmmC with a Web application as depicted in Figure 6.32. Apart from using the same annotation types described for XmmC, we experimented with text annotations, where the content can be seen and edited in real-time. This component uses an Operational Transformation

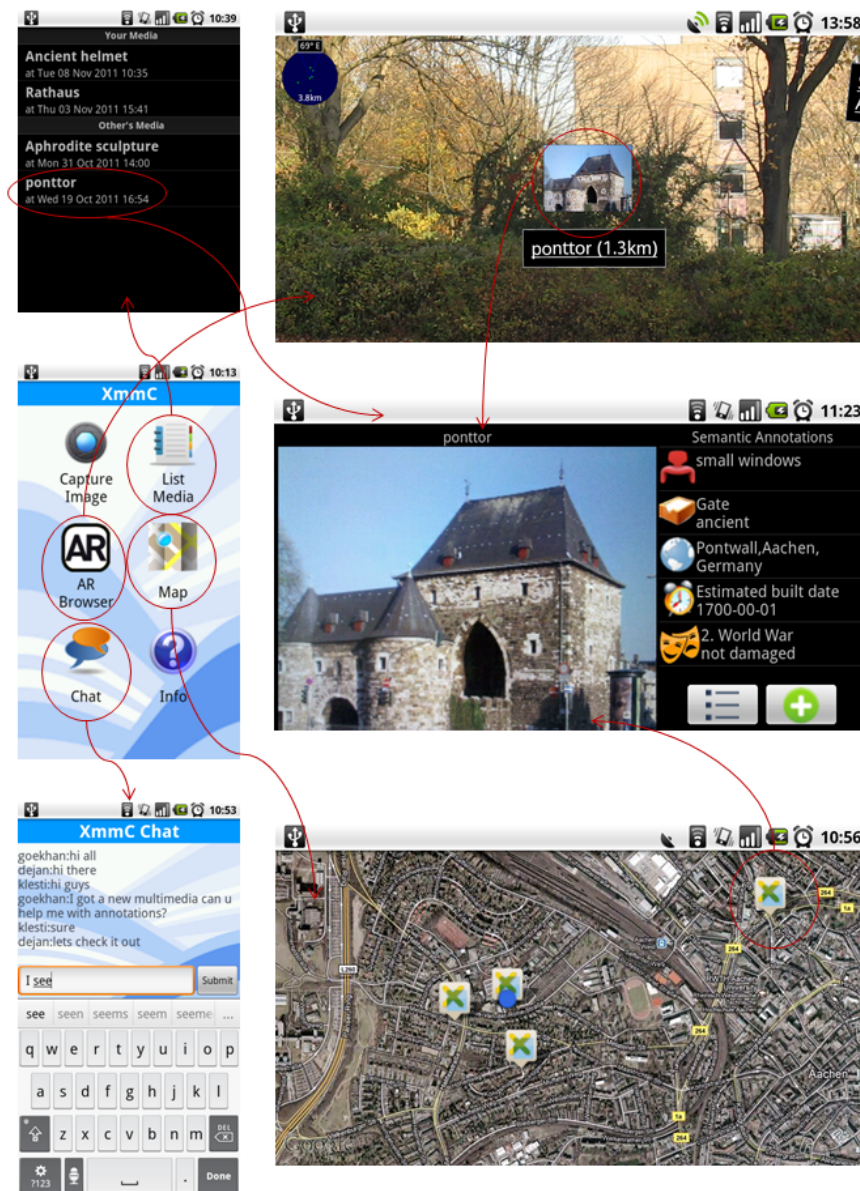


Figure 6.31: UI sequence of actions for collaborative annotation

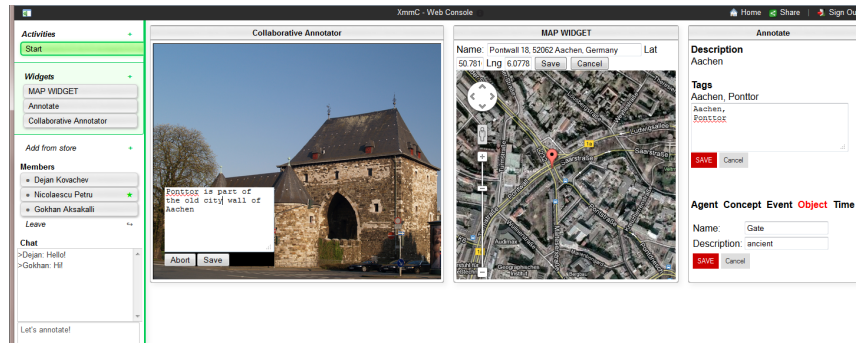


Figure 6.32: Web application for collaborative annotation

```
<iq type='get' id="XcupS-1"
  from='alice@merian.informatik.rwth-aachen.de/Smack'
  to='xmmc@merian.informatik.rwth-aachen.de/Smack'>
  <ar-query
    xmlns='http://dbis.rwth-aachen.de/xmmc#services/AR'>
    <latitude>50.77816836</latitude>
    <longitude>6.0616060</longitude>
    <radius>800</radius>
  </ar-query>
</iq>
```

Listing 6.3: AR-query Get IQ Stanza Example

engine and XMPP over Web sockets as a communication protocol for text editing operations and updates. The Web interface is developed using Web widgets (components with limited, but clear-cut functionality). We use a XMPP publish-subscribe mechanism to dispatch messages between the widgets, technique which allows communication of data across browser instances. Through this Web extension, we offer a better support for stationary users, which are collaborating to remote on-site users, using XmmC on their mobile devices. Moreover, we test the feasibility of managing the XmmC content in a Web browser and the impact of adding the real-time editing feature to the overall annotation experience. However, due to space reasons, more related technical details are not provided here.

Listing 6.2.2 illustrates a get ar-query custom IQ stanza with an example. The client side issues a get IQ ar-query stanza with the location restriction parameters latitude, longitude and radius that conforms to a circular geographical area with a center point of the circle is denoted by latitude, longitude and radius of the circle denoted by the radius.

After the AR Module receives the request it retrieves the POIs which are inside the restriction area and sends back a result IQ stanza with a ARML element composes of POIs that provides basic information. Listing 6.2.2 shows result ar-query IQ stanza.

Figure 6.27 shows that multiple XmmC clients can communicate with a XmmC server over

```
<iq type ='result' id="XcupS-1"
  from ='xmmc@merian.informatik.rwth-aachen.de/Smack'
  to ='alice@merian.informatik.rwth-aachen.de/Smack'>
  <ar-query
    xmlns='http://dbis.rwth-aachen.de/xmmc#services/AR'>
    <ar-arml xmlns:ar="http://www.openarm1.org/arm1/1.0"
      xmlns:wiktitude="http://www.openarm1.org/wiktitude/1.0">
      <ar:provider id="dbis.rwth-aachen.de/XMMC">
        <ar:name>XMMC</ar:name>
        <ar:description>
          XMPP-based Multimedia Collaboration
        </ar:description>
        <wiktitude:providerUrl>
          http://dbis.rwth-aachen.de/~goekhan/xmmc/
        </wiktitude:providerUrl>
        <wiktitude:tags>
          media, real-time, collaboration
        </wiktitude:tags>
        </ar:provider>
        <ar-placemark id="16a52a60ffa0.jpg">
          <ar:provider>dbis.rwth-aachen.de/XMMC</ar:provider>
          <name>Battleship Averof</name>
          <description>
            with 3D animation of battleship in aciton
          </description>
          <wiktitude:info>
            </wiktitude:thumbnail>
            http://.../MediaRepository/files/16a52a60ffa0-thmb.jpg
            </wiktitude:thumbnail>
            <wiktitude:url>
            http://.../MediaRepository/files/16a52a60ffa0-thmb.jpg
            </wiktitude:url>
          </wiktitude:info>
          <Point>
            <coordinates>6.1334,50.5036,12</coordinates>
          </Point>
        </ar-placemark>
        <ar-placemark id="3b90aa319d44.jpg">
          ...
          ...
        </ar-placemark>
      </ar-arml>
    </ar-query>
  </iq>
```

Listing 6.4: AR-query Result IQ Stanza Example

the XMPP protocol. The client application requires mobile devices with GPS, compass, accelerometer, WLAN and camera. Android was selected as mobile software platform for this prototype. It allows many already developed Java libraries or Java projects to be imported into the Android applications.

XMPP is the main communication protocol used in this system. Since, Smack [Real11a] is most mature XMPP client library written in Java, it is used for XMPP communication with the XMPP server. The XMPP connection layer is responsible for receiving and sending XML stanzas by using the Smack library. CEFX component is used for synchronization of the XML metadata. The Android application operates both as a MAR browser and provides

features to users for media acquisition and metadata annotation.

On the server side the XMPP communication with clients is enabled by an XMPP Server [Real11b]. The XmmC server actually acts as an XMPP client considered from the XMPP Server. This simplifies the software development and maintenance since both the mobile client and XmmC server can use the same communication components built atop Smack.

Evaluation

We evaluated the system in terms of performance and user experience in a mobile environment. The evaluation process was separated in two main parts. i.e. technical evaluation and user experience evaluation.

Contemporary real-time collaboration systems run well and consistent with Web-based or desktop clients. However, ensuring real-time responsiveness within mobile network settings can be problematic due to the unstable, low-bandwidth, high-latency mobile network connections. In this part, we evaluate the XMPP-based CEFX framework for collaborative annotation context in mobile settings. We can divide evaluation process of this section into two parts. First, performance test for sending and executing remote updates were conducted and the results were analyzed. Second, we examined the framework for conflict resolution and consistency maintenance in simulated mobile network settings.

In the context of performance test, the test suit from the previous section is used but now with two mobile devices. We measured the time passed during a mobile client's generation of an operation with adding, modifying or deleting a semantic annotation and sending the generated operation to the other mobile client until corresponding operation is executed at the client. In the test scenario, the client prototype has generated 10 insert, 10 update and 10 delete operations.

The collected performance values are demonstrated on Figure 6.33. The average time was 412 ms with standard deviation of 209 ms based on on the collected performance values. Both average and deviation values are acceptable in a mobile real-time collaboration scenario. However, these values depend on network characteristics too. We have done our tests with a WLAN connection, the process times can be longer with different mobile networks like GPRS, EDGE, UMTS, or LTE.

We also examined the framework for conflict resolution and consistency maintenance at annotation scenario in low bandwidth high-latency networks. In order to provide such network settings, we used the Android emulator. The emulator was set up with network settings of 900 ms fixed latency and 1.2 KB/s download and upload speed. The test setups consisted of client prototypes on the emulator and on a mobile device from previous tests. First, we tried to delete the same annotation concurrently from both clients. Both operations were sent to the server however, the operation from emulator arrives to the server after the operation of mobile device was executed. The framework resolves the issue by discarding the message from the emulator. Another possible conflict, is to modify the annotation, which

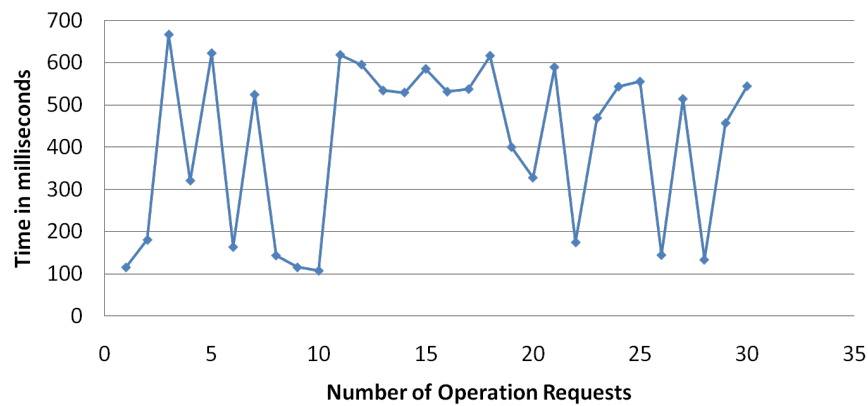


Figure 6.33: Execution time of remote concurrent XML editing operations

is also concurrently deleted. Emulator client tried to modify an annotation, concurrently device client tried to delete an annotation. First delete operation was reached to the server then the modify operation. Server executed first the modify operation then the delete operation. We also tried to modify the same semantic base type from both the emulator and the device concurrently. In this case the latest operation effect is seen on the server side. However, the CEFX+ framework does not always guarantee convergence and intention preservation since Android and server implementations uses different DOM level versions, it can lead to exceptions on the Android site and this is still an open issue.

Energy efficiency is a fundamental consideration for mobile devices due to the short battery live issues. As you may recall, we have chosen XMPP as a general communication protocol. However, XMPP was mainly originated for instant messaging and mobile platform considerations were not taken into account. Due to the nature of the protocol, XMPP generates verbose XML streams, which is acceptable in general, however it shortens battery life in mobile environment by over-generating network traffic. Moreover, XMPP requires a constant TCP connection. Open network sockets are very energy consumptive [Kore08]. Moreover, network outages or connection type changes in the underlying mobile network connection, e.g. WLAN to GPRS, cause to repeat all network socket and XMPP session establishment process.

In order to analyze the reason of verbose XML streams, we had logged the XMPP stanzas at the XMPP server. We observed half of the traffic as functional stanzas and the other half consists of available and unavailable presence messages. High percentage of unavailable messages can be explained by the unstable characteristic of the mobile connection.

In cultural heritage scenarios, it is expected to view the POIs that represents acquired and annotated multimedia at right geographical locations. We tested the accuracy of perceived POIs' location. First, ten artifacts were acquired and their geographical location was edited so that they have different distances and directions to the current location. Afterwards, AR browser was tested outdoors to observe the perception of where POIs are appeared. We conducted the test in outdoor intentionally in order to retrieve more accurate current location

information by making use of GPS signal. We observed that the location perception of POIs were accurate enough for long distance POIs however, it was not accurate for the nearby POIs.

We can explain the results by examining how AR browser calculates the location of POIs. AR browser highly relies on the GPS to get current location, compass and accelerometer to indicate pitch and roll of the device. However, these sensors doesn't provide 100% percent accurate data. Especially, compass can generate incorrect data due to the presence of nearby metal objects.

A user study and performance test we conducted. Seven participants tested the mobile services for collaborative metadata annotation. Different Android smartphones and tablets were used. The OS version ranged between Android 2.2 and 3.1. The devices' hardware ranged between 400MHz and (dual-core) 1.2GHz CPUs and between 256MB and 1GB RAM memory. In order to enable participants to assess the different qualities of the services, they completed short-term tasks that simulated documentation work at a cultural heritage site. The tasks consisted of multimedia acquisition and sharing, context-aware metadata creation and collaborative annotations. The goal of the user study was to evaluate the context-aware mobile collaborative services delivered from the i5Cloud.

The role of the prototype was expressed as a technical tool for digitally documenting the historical sites with semantic annotations and increasing the cultural heritage awareness. Since, it is significant to get actual feedback from users that tried achieve these goals, we conducted a evaluation session with seven participants.

In the scope of the evaluation session, we came across with some challenges. First, conducting the evaluation session at a real historical site would be ideal, however most of the participants do not have UMTS/3G connection as seen at Table 6.7. Therefore, we tried to simulate the historical site by mapping historical artifacts at Aachen area as shown at figure 6.34. Another challenge is the number of participants. The number of evaluators is limited to seven, since we have restricted number of mobile devices. It would be better to have more users to demonstrate scalability of the system, on the other hand but seven participants is enough to visualize the concepts at the cultural heritage scenario.

Instruction guidelines were handed out to the evaluators before the start of the evaluation session. Once all evaluators logged in to the application, they received a short briefing about the basic features of the application. After that, the participants were requested to the perform the following tasks.

- **Multimedia Acquisition** Users were asked to capture corresponding image and upload it to the system.
- **Annotating the Multimedia** After acquisition, the users were asked to annotate the multimedia together with the help of these documents. They were also expected to make use of chat conference feature in order to organize annotation process.

- **Augmented Reality** The evaluators were divided into 2 groups and they situated themselves at different locations. They were told to imagine their surroundings as a historical site and acquire interesting things and annotate them together as the previous parts. Furthermore, the evaluators used AR Camera and Map views to browse multimedia. They were also asked to compare geographical location consistency of the multimedia at Map and Camera View.

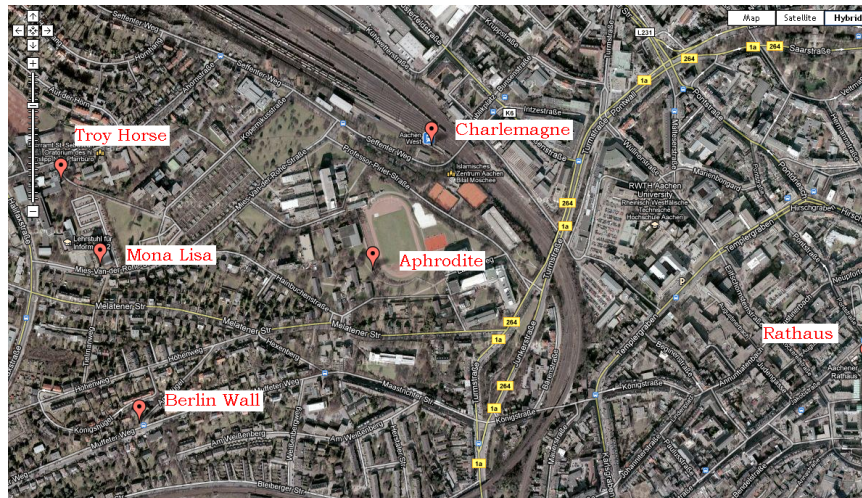


Figure 6.34: Location mapping of historical artifacts at evaluation session

At the end of the evaluation session participants were requested to fill a questionnaire in order to get a feedback.

Results and Discussion

Participants at the evaluation session had different devices with different Android OS versions.

Dev. Model	Android Version	Network	Processor	Screen Size
Samsung Galaxy Tab 10.1 Tablet	3.1	WLAN	1 GHz Dual Core	10.1"
Motorola Defy	2.2	UMTS 3G	800 MHz	3.7"
HTC Desire	2.2	UMTS 3G	1 GHz	3.7"
Samsung Galaxy S2	2.3.3	WLAN	1.2 GHz Dual Core	4.3"
Motorola Milestone	2.2	WLAN	550 MHz	3.7"
Samsung Galaxy S	2.3.4	WLAN	1 GHz	4.0"
HTC Desire HD	2.3.3	UMTS 3G	1 GHz	4.3"

Table 6.7: Mobile devices used at XMMC evaluation session

We can demonstrate the fact that even the participants who had previous knowledge about the mobile multimedia annotation can still learn new things and increase their cultural heritage awareness. This goal was achieved as seen at Figure 6.35.

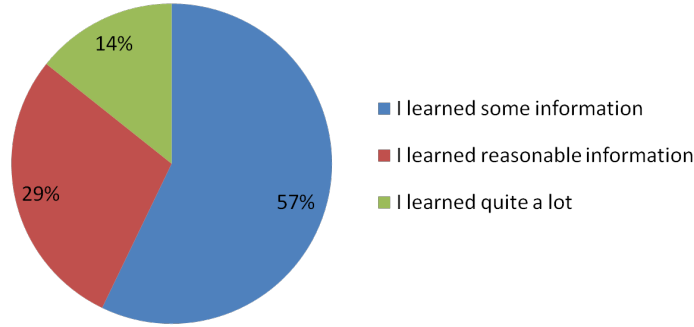


Figure 6.35: Increase of cultural heritage awareness

Evaluators were mostly satisfied with the user interface. They also found it responsive and user friendly as seen at Table 6.8. They sometimes used chat functionality to get better organized for annotation and all evaluators find chat functionality useful. During the collaborative annotation the majority of the participants (5) perceived the real-time updates on the annotations however some of them (2) couldn't. This can be understandable because in order to see real-time updates, at least one other user should also annotate the same multimedia at the same time.

The outdoor AR experience is significant to evaluate the accuracy of the AR browsing using the Camera and Map view. As expected, the Map view shows the locations at the right positions. However, the questionnaire feedback pointed out that the AR camera browser provided less accurate geographical location perception in certain cases. The participants gave subjective ratings on how consistent were the perceived geographical locations of POIs at Camera AR view and the actual Map view. The results showed mean subjective accuracy of 77% with standard deviation of 12%.

Question	mean	std. dev.
XmmC UI is user friendly?	4.29	0.45
XmmC UI is responsive?	4.29	0.45
Adapting media accuracy is meaningful?	4.14	0.64
Integrated chat is useful for collab.?	4.86	0.35
I received real-time annotation updates?	4.43	0.73
Provided reverse geocoding is useful?	4.71	0.45
Filtering POI is useful?	4.00	1.07
I have noticed real-time POI updates?	4.14	1.12
How useful is the complete system?	4.71	0.45

Table 6.8: XMMC user evaluation questionnaire results

XmmC is also capable of integrating with LAS MPEG-7 multimedia services. We also compared the basic metadata information and semantic annotations on the XmmC client

and MPEG-7 services. The results showed that, XmmC propagated all annotation requests to LAS MPEG-7 services successfully.

In the context of the evaluation session, 7 participants acquired 13 multimedia artifacts and they created and modified 47 semantic annotations collaboratively. All of the evaluators found the application successful and useful in general.

We evaluated the implemented system based on technical and user point of view. In the technical part, we tested the prototype for storing and transferring for various media types. Then, the feasibility of XMPP-based CEFX+ framework in a mobile collaborative annotation scenario was evaluated for real-time performance and consistency maintenance. We also compared battery consumption while performing various tasks and examined XMPP protocol for energy efficiency. Technical evaluation finalized with examining the AR browser's accuracy to show the POIs at the right position. In the user experience based evaluation part, we conducted an evaluation session to examine XmmC while evaluators used the prototype for collaboratively documenting a historical site and increasing their cultural heritage scenario. After the evaluation we compared the acquired multimedia artifacts at XmmC and LAS MPEG-7 multimedia services.

In the light of these results, we can conjecture that XmmC is serving its purpose. However, a larger scale evaluation on more historical sites by professionals will allow us to improve the prototype.

6.3 Personal Cloud Computing

Personal cloud computing as a specific area of cloud computing is currently gaining momentum. The *personal cloud* allows consumers to seamlessly store, sync, stream and share using multiple connected devices such as smartphones, media tablets, televisions and PCs over the Internet. The emergence of personal clouds reflects the “4S experience”, consumers’ desire to store, synchronize, stream, and share their content on regardless of device or platform seamlessly. A personal cloud is the location where users store their personal content and preferences and where there are able to access personalized services. The web of personal devices is being used in different context of our daily lives. Consequently, there is a shift of the focus from a user’s device to cloud-based services delivered across multiple devices. The specifics of devices become less important, i.e. users will use a collection of devices, without having one device as a primary hub. Personal cloud services cannot be tied to one specific device or platform. An invisible experience means it works on anything. Service providers must deliver an invisible content synchronization experience between: device to device, person to person, screen to screen, and location to location.

Vander Wal [Wal05] discerns four types of information clouds:

- Global InfoCloud: the Internet, with plethora of information, but often too much information and users are flooded with information choices. There is a little heed given to how the user could consume and reuse that information.

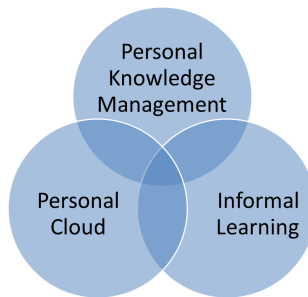


Figure 6.36: Main aspects of personal clouds for learning

- Local InfoCloud: Information that is contained in community wikis, walled information systems, location-based databases.
- External InfoCloud: The information that is need for a user but it is outside his/her reach.
- Personal InfoCloud: Collected information that follows the user. People have their own needs, so this information cloud is highly personalized with services to reuse and re-find information.

Typical public personal clouds provide simple services such as document, photo, notes and content sharing between devices, i.e. specialized multimedia services.

6.3.1 Learn-as-you-go: Personal Clouds for Learning

Nowadays, mobile and web technologies are developing new ways for learning and knowledge attaining. People use Web 2.0 platforms such as Wikipedia and (micro-)blogs additionally to augment the existing course materials for formal learning and education. Furthermore, smartphone applications and mobile Web make learning content be well accessible anywhere at any time. The convergence of the Web and mobile platforms currently is an enabling factor for novel informal learning methods such as micro-learning.

Personal clouds allows learners to expand their capacity of knowledge by linking to external resources such as online databases, experts, peer learners, or content management systems. Learners that are able to develop skills to search and find relevant information on the Web, organize and manage links to the information, are also able to extend their knowledge beyond their brain capacity [McBe10].

Figure 6.36 shows the main aspects of personal clouds for learning. Beside the personal cloud concept which is described previously, personal knowledge management (PKM) techniques and informal learning strategies play important roles. PKM is a collection of processes that a person uses to gather, classify, store, search, retrieve, and share knowledge in his or her daily activities. PKM also refers to the way these processes support work activities.

PCL can facilitate different styles of learning as we all approach learning in different ways. Personal learning has proven to be more efficient and effective for various learning objectives in different education settings. Personal learning relates to the creation of learning content by using it and reflecting about it. The content has meaning for the learner because it includes cognition, emotion and motivation. Frequently, learners participate in informal learning processes beside the formal instruction received in education institutions. Informal learning occurs outside the education establishments — it happens sporadically and accidentally. Seventy percent of what people know about their jobs, they learn informally from the people with whom they work [Cofe00]. Moreover, successful learning is a constructive process that involves seeking solutions to problems and relating experiences to existing knowledge [STVa05]. Sensors and displays, and mobile devices play an important role in designing this continuous cycle of experimentation and reflection.

Personal clouds for learning (PCL) are not a software application, but an approach of using technologies for learning. Whereas in other technology-enhanced learning concepts like personal learning environments (PLE) the focus is on the pedagogic, ethical and philosophical side, in PCL the focus is on the technological side. PCL doesn't follow any design or learning process. Basic PCL enable learners to easily organize, discover, categorize, upload, share and most importantly reuse any media content on the Web on any personal computing device. Learners, instructors and institutions are free to use any tools, and any design and learning process. Education is provided on-demand, which makes PCL appropriate for life-long learning and for forming learning networks that are not institutional.

Micro-learning

Micro-learning is an example of technology and design driven pedagogy. It demonstrates many aspects of personal clouds applied to learning. In principle, “microlearning” is pedagogically agnostic. The crucial challenge is the seamless and organic integration into the daily digital workflows and life streams [Hug05].

Micro-learning refers originally to taking short-term-focused learning activities on small learning content units [Hug05]. We refine micro-learning as a learning activity on small pieces of knowledge based on web resources. Micro-learning differs from micro-blogging (e.g. Twitter) in way that the later is more about disseminating information (a single resource) to other people, and the former is about collecting personally relevant information (several resources) from many sources and using that collection to cover personal knowledge gaps. Micro-learning consists of a fast, convenient and instant capture of the self-identified knowledge gaps, understanding them with the help of online resources, creation of a learning object out of these online resources and integration of that learning object into small learning activities interwoven into our daily life.

Our research on micro-learning is motivated by the following considerations. First, exploiting the ever growing Web content as a primary source for personal knowledge enrichment requires usage of many independent online services to acquire the relevant information. A

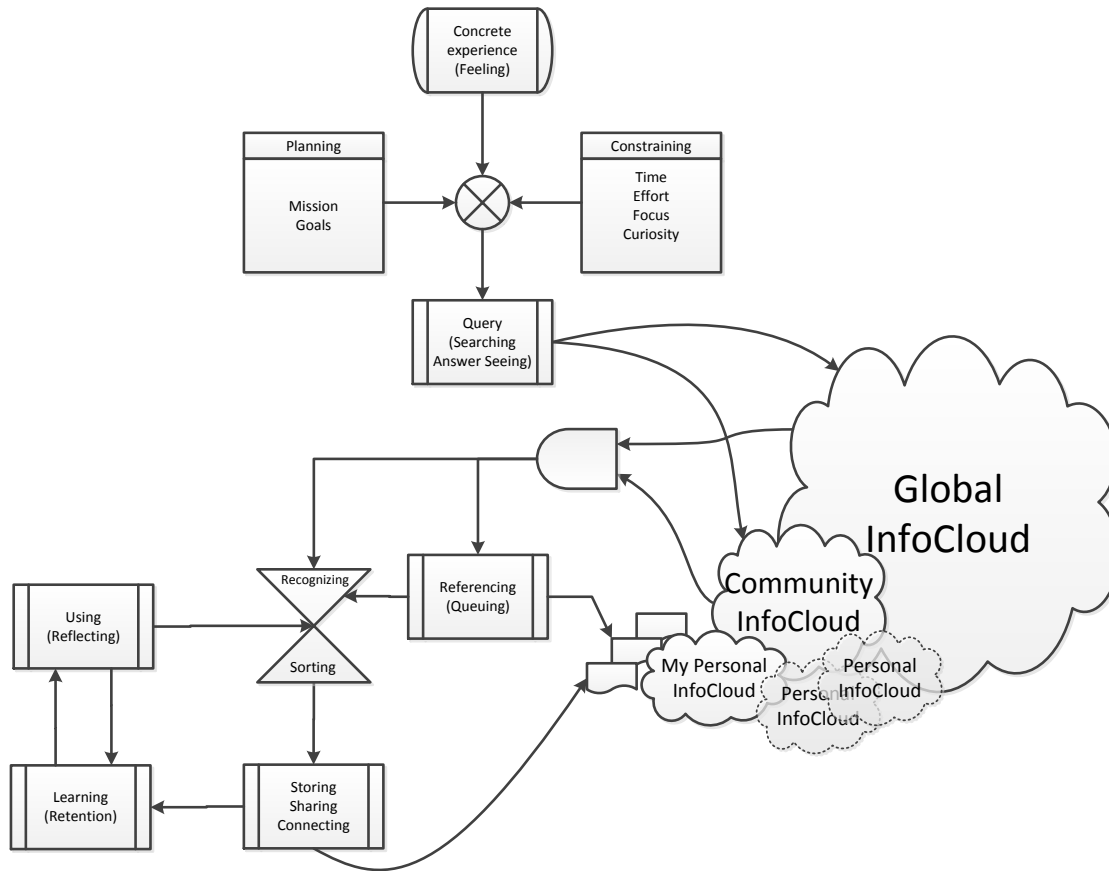


Figure 6.37: Micro-learning information life-cycle

combination of heterogeneous digital resources or fragments is required. Second, personal learning content is a subject to a “evolution”. Many of the learning systems consider relatively static learning content with less adaptiveness. But in case of micro-learning, learning content changes constantly. Users enrich learning content with new web resources. Third, the learning phases (plan, learn and reflect) in micro-learning span over different spatio-temporal settings. There is a disparity of the periods (and available devices) when people encounter the knowledge gaps and when they have time to learn and reflect. Therefore, different devices should be supported. Fourth, switching context from the primary activity to a learning process causes distraction from the main activities. People should be able to acquire the information artifacts quickly and learn them at suitable time, without much interruption. Fifth, since micro-learning involves heterogeneous content, pre-defined data models and learning processes would not fit. Tagging offers a simple but yet powerful mechanism to organize, save and distribute learning content and to regulate personal learning processes [HNWo08].

In order to support learners’ micro-learning activities, we conceptualize a micro-learning

model with three technical foci. We have developed tools for ubiquitous acquisition of digital learning resources both on a desktop web browser and a smartphone. Furthermore, we apply cloud-based approaches to micro-learning synchronized between Web and mobile learning environments. Cloud technologies are applied to augment and synchronize data processing capacities on smartphones [KCK110]. Finally, we use tags for micro-learning in different learning phases based on the tagging model in our previous research [CKK110]. This tagging concept is proposed to meet challenges in learning resource management and learning process support. As a proof of concept we have applied the implemented tools in the use case of vocabulary learning.

The information life-cycle on Figure 6.37 depicts the certain micro-learning activities in relation with the information clouds.

Ubiquitous Learning Content Creation and Enhancement

Ubiquitous communication and computing technologies enable smartphones to be widely used to acquire instant knowledge anywhere anytime. They are good for instant learning content acquisition (images, videos, sound, location), but devices with bigger displays (laptops, desktops) are more convenient for enhancement and completion of the learning objects from a learners' perspective. Therefore, we exploit smartphones for instant context-enriched content acquisition and as tools for informal learning, and desktop web browsers with extensions to manage and enrich personal learning content.

The computing clouds play a significant role in gluing the heterogeneous parts of the system, in order to provide unified learning experience. Each micro-learning activity is initiated within some concrete context, e.g. searching for a definition of some term. Users nowadays usually research on the Internet—the largest and most convenient collection of information. In many cases, user's information needs cannot be immediately answered by a single web page or search results page, because these needs are complex and highly-personalized. This typically occurs when users query for certain topic in domains like learning, travel, health, where users need to consult with multiple heterogeneous web sources. For example, searching for meaning of some phrase in German for non-native speaker involves looking up in multiple translation, dictionary, thesaurus and example web sources. Donato et al. [DBCM10] showed that these “research missions” account for about 25% of the search volume on Yahoo! Search.

Each lookup, i.e. research mission, causes time consumption and mental efforts. Therefore, it deserves special attention and treatment. Federated search has been proposed in the research literature as a remedy. However, most of the proposed tools tend to be efficient only in certain closed domains such as flight ticket or hotel searching. We propose a browser extension called Multi-Lookup where the user can specify the sources for web search. The user can then join the individual sources (i.e. search engines) into groups which try provide comprehensive results for the intended search interest. This tool integrates within the browser search functionality. The user needs to specify which search group, he/she

wants to use for the next query. The result pages are opened in reusable tabs in the browser, i.e. subsequent queries use the already-opened browser tabs.

Mobile OCR Input

Novel methods like Optical Character Recognition (OCR) help alleviate some of the inherited issues of their small computing devices. OCR refers to the process for acquisition of text and layout information, through analyzing and processing of image files. Compared to the traditional input way of typing, OCR technique has many advantages such as speed and high efficiency for large texts. Furthermore, in the case of Asian or Arabic alphabets OCR eases the input of characters and words. Several mobile products are trying to commercialize this convenient approach⁶.

In our system the OCR technique is used as a tool to catch the source words from document and we implement this component based on the open-source project “Wordsnap”⁷. This input method allows users to input the words they want to query or learn in a more convenient way when they read printed documents.

Web Content Scraping

After identifying several result pages as potential answers to the search query, the user tries to identify a comprehensible answer. Our Web content scraping tools enables users to collect relevant pieces of information from these result pages. It features visual highlighting of web page components (paragraphs, divs, articles, flash objects, images, headers, links, etc.). Parts of different Web pages are selected by the user that he/she thinks they are most relevant in the current context. These parts (clips) are appended to the learning object (note). This tool reduces the time needed to collect chunks of Web pages since it reduces the selection operation to a single click on highlighted elements. Moreover, the collected clips are shown in the browser sidebar, thus preventing the user of losing focus when switching between windows and tabs.

Our Web browser add-on uses client side DOM-parsing Web scraper to ease the selection of content fragments from multiple web pages. The tool highlights the information chunks on a Web page and lets the user select only relevant parts (text, paragraphs, divs, images, videos, etc.) out of it. Figure 6.38 shows how parts of different Web pages were selected by the user that he/she thinks they were most relevant in the current context, and were added to the learning object. The user can then use other Web pages to enrich the learning object, e.g. images from Flickr or videos from YouTube, in order to make it more understandable and explanatory. More specifically, the user inputs difficult-to-type Chinese characters via OCR, thus initializing a learning object which is synchronized to the cloud (top left). The browser

⁶<http://www.pleco.com/>

⁷<http://www.bitquill.net/trac/wiki/Android/OCR>

add-on (down left) helps getting relevant content from several web sites (down right). The augmented object is synced back to the cloud. The user can learn using rich-content learning objects on the mobile device (top right).

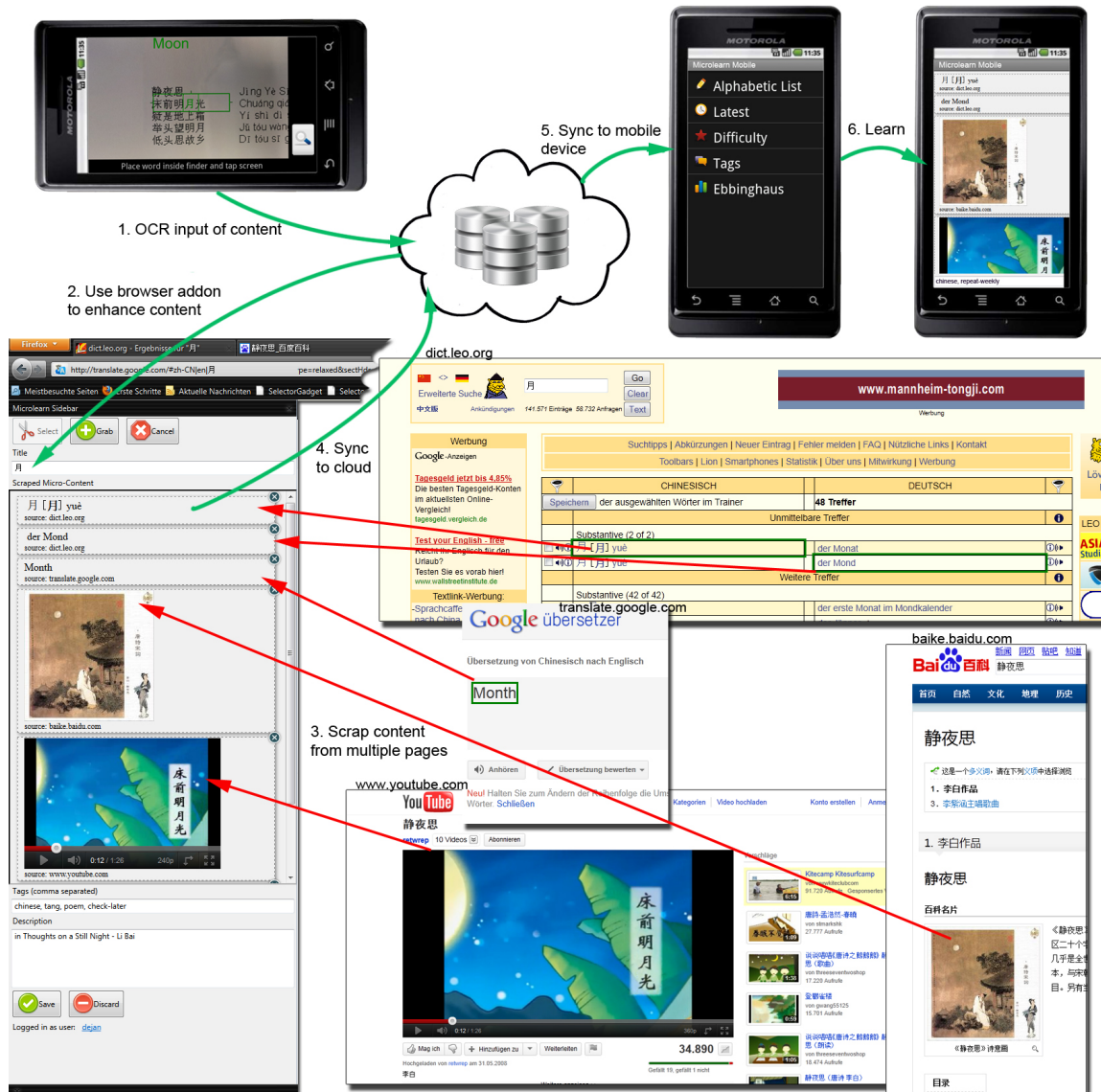


Figure 6.38: Micro-learn workflow

Mobile Learning App

The prototypical implementation and testing of the mobile learning application were supported by a bachelor thesis [Bran12]. The application is divided into two different modes - browse and learn modes. The browse mode provides functions for creating, editing and

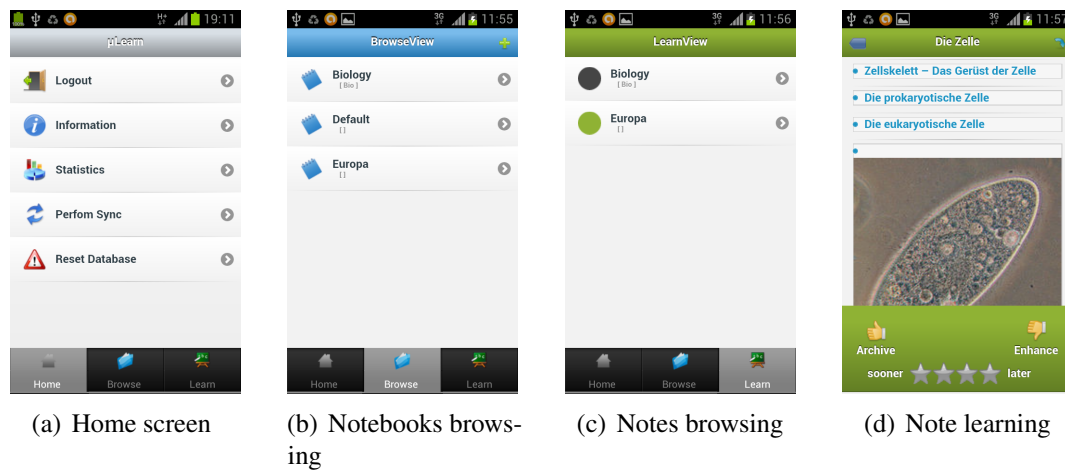


Figure 6.39: Micro-learn evaluation results

gathering information. The user can sort the notebooks by different criteria (see Figure 6.39(b)). It is also possible to search the complete content for keywords. This increases the usability and makes finding notebooks or objects easy even if the user has a lot of learning objects. It is possible to edit every notebook, note and clip. In the learn mode the user can learn the gathered information by selecting the notebook he/she wants to learn (see Figure 6.39(c)).

Cloud-based Data Synchronization

One of the biggest challenges in multimedia application development is device heterogeneity. Users are likely to own many types of devices. Switching from one device to another, users would expect to have ubiquitous access to their multimedia content. Cloud computing is a promising solution to managing heterogeneous learning content, i.e. images, text, web pages excerpts, links, videos, etc. and delivery of computing and storage services as utility.

In our case, the cloud features a personal learning content vault for each user. It handles data storage, data processing, and adaptive content delivery to different devices. These cloud learning services run on top of the scalable multimedia cloud infrastructure, which is more described in our previous research work [KoK10].

In general, in informal learning the acquisition and learning phases happen on different devices. For example, unknown phrases are heard during conversation and recorded on the smartphone, and further understanding is carried on a desktop or laptop computer. Therefore, a seamless integration of web-based tools with mobile applications is required for the purpose of enabling learning anywhere at any time [LTLi10]. The content and metadata both should be up-to-date using data synchronization technologies.

In the whole system design smartphones are not the only way to add data to the personal

learning vault, i.e. it is exposed through a REST API for other clients to access and change the data. However, providing such access requires methods for synchronization of the data between all clients. For better network traffic efficiency, an incremental synchronization technique is implemented. The small mobile devices have a local partial replica of the user's learning content synchronized with the cloud personal learning content vault which is also accessible from the Web.

Evaluation

We developed an Android application that features translation, OCR input of text, tagging of learning content, reinforcement learning model and synchronization with the cloud. Furthermore, we developed a browser add-on that enables users to enhance the learning objects with rich multimedia content and to further refine the learning content management.

In order to obtain a comprehensive evaluation of the prototype and get the result of the users' feedback to summarize advantages and disadvantages, referring to Chen et al. [CLCh07], we have adopted a series of simplified questions of system operations to measure if these tools is able to satisfy the needs of users. Ten students who studied more than one foreign language were asked to use the tool for a period of time and to use it for bilingual vocabulary learning. Only 2 out of the 10 students had used a smartphone for learning purposes before. The students then answered a questionnaire.

The evaluation of the current version revealed positive feedback for supporting bilingual vocabulary learning, synchronization of learning content, usage of online resources for learning content creation and enhancement, and usage of multiple strategies for vocabulary learning (time, tags, language, difficulty, Ebbinghaus). The drawbacks of the system are slower OCR recognition than typing for Latin script languages, issues with color printouts (but for Asian scripts the value of using OCR input is huge). With the development of the OCR technology we could improve the function of OCR to get a more powerful learning tool. Currently, some applications depended on object recognition from images have already emerged such as Google Goggles.

In addition, the mobile learning application was evaluated to get to know whether the application is easy to understand and intuitive to use. To be able to analyze the user evaluation in an appropriate way, a structured questionnaire was created. The questionnaire consists of questions regarding the design, the usability of the application and some information about the user. The application and the questionnaire were handed to six test users. Both experienced and non experienced users were chosen. After one week the filled out questionnaire was collected and evaluated. The focus of the user evaluation is the usability of the application and not the learning effect. The test user should focus on using the application. So some example content and instruction manual were created. It included a video, where the user could see how the content is selected from the Web.

The test users were intermediate to expert smartphone users and most users had an age between 16 and 25. The age group of 26-35 and 46+ was covered, too. Every user had a

different device, so the application was tested on six different devices with also different OS versions. Most of the users used the application 10 to 20 times in the given time frame. All users understand the application and even the meaning of the most icons was clear. The appearance of the “skip” button caused some trouble. All participants explained the differences between browse and learn view correct. The results show that the design and the structure of the app are intuitive, the test users had no problems with using the application. Also the user understood the meaning of the different views. The used icons are intuitive to use, except the “cancel” and “skip” button. These button were redesigned to a button which contains text.

6.3.2 DireWolf: Distributed Web Interfaces for Device Clouds

Web applications have overcome traditional desktop applications especially in collaborative settings. However, the bulk of Web applications still follow the “single user on a single device” computing model. Therefore, we created the DireWolf framework for rich Web applications with *distributed user interfaces* (DUI) over a federation of heterogeneous commodity devices supporting modern Web browsers such as laptops, smart phones and tablet computers. The prototypical implementation and testing of DireWolf components were supported by a master thesis [Li13].

The DUIs are based on widget technology coupled with cross-platform inter-widget communication and seamless session mobility. Inter-widget communication technologies connect the widgets and enable real-time collaborative applications as well as runtime migration in our framework. We show that the DireWolf framework facilitates the use case of collaborative semantic video annotation. For a single user it provides more flexible control over different parts of an application by enabling the simultaneous use of smart phones, tablets and computers. The work presented opens the way for creating distributed Web applications which can access device specific functionalities such as multi-touch, text input, etc. in a federated and usable manner.

Emergence of Personal Device Clouds

People increasingly interact with a collection of heterogeneous computing devices attached to their daily lives. However, most Web applications fail to combine devices’ features into a cohesive symbiotic way to convey a single user task in a collaborative fashion. One of the reasons behind this failure is the lack of tools and methodologies required to develop applications spreading user interfaces across multiple devices available to a particular user or group of users. Personal computing is no longer confined to a single device. People, on average, have more than one personal computing device which are used in different contexts. In certain circumstances people prefer to use their mobile devices rather than desktops or notebooks, e.g. smartphones are used for voice communication and tablets provide a more natural multi-touch screen functionality. PCs together with commodity

smartphones, tablets, eBook readers, gaming consoles and interactive TVs can be federated over the Internet to create collaborative multi-device interactive systems which can benefit from the diverse device capabilities. An individual can interact in different ways with such symbiotic computing environments, consisting of personal devices.

As a consequence, monolithic single-device *user interfaces* (UI) devolve to Distributed User Interfaces. DUIs separate, migrate and merge seamlessly between devices. Additionally, they can adapt to different platforms [LGL*11] and account for changes in device availability to achieve a continuous application experience [V VLC05].

Developing distributed user interfaces is challenging [BRA11]. From the user perspective, two challenges are salient. First, users should be supported to adapt the distribution to their needs. Second, users should experience seamless UI migration. Migrated UI components preserve state and remain consistent with the whole application context. Concerning the use of multiple devices, current Web applications can be well rendered on different platforms. However, most of them ignore the possibility of using multiple personal computing devices. Cooperation between such devices related to distributed interfaces is scarce and mostly limited to device-specific static interface separation.



Figure 6.40: An example of distribution of user interface components (widgets) to diverse (mobile) computing devices

To address these challenges, we developed DireWolf, a framework for distributed Web applications based on widgets. We have chosen to work with Web widgets because they represent interface components with limited, but clear-cut functionality, dedicated to smaller tasks. Widgets can be shared, reused, mashed up and personalized between applications. By splitting the interface into separate widgets and enabling them to exchange information, customizable Web applications can be developed. Whereas previous work [BSGi11, DST*12] on widget applications and mashups considers single-end devices only, we examine the concept of widget-based Web applications combined with device awareness, session mobility and cross-device cooperation.

To illustrate the concept, we shortly describe a semantic video annotation scenario with SeViAnno (cf. Figure 6.40). SeViAnno application was transformed from a typical Web application into a widget-based one, thus validating the feasibility of our approach. A semantic video annotation application is an ideal candidate for extended UI interactions: users watch videos, annotate them at certain time points or for specific time intervals and navigate through a video using the annotations. Various types of available semantic annotations (agent, time, concept, object type) can be added using text input and interacting with a video player. Place annotations can be pinpointed on a map. However, e.g. full screen mode of the video player hides all other UI controls on one device. In an annotation scenario, distributing the UI enhances user experience. Users can play the video in full screen on one device and can use additional devices to annotate it or to browse through the video. Moreover, they can use device-specific features for each of the UI elements, e.g. multi-touch on a smartphone for interacting with a digital map. Preserving UI state across devices is also required for such a scenario, e.g. resume at current position instead of restart after migration of a video player, continue annotating, etc. The DireWolf framework brings forward the following contributions:

- a framework for easy browser-based distribution of Web widgets between multiple devices
- facilitation of extended multi-modal real-time interactions on a federation of personal computing devices
- provision of continuous state-preserving widget migration

The technical realization ensures real-time awareness of the user's personal devices availability and supports *UI partition*, *UI component mobility* and *device cooperation*. Web applications can entirely or partially be distributed to the available devices. The framework provides the Web application a flexible and configurable UI component distribution at runtime. DireWolf helps managing a set of devices and handles communication and control of distributed parts of the Web application. The conceptual and implementation details of the DireWolf framework, together with the possibility of integration into existing widget platforms is detailed in the next sections.

DUI Approaches

The migration of a user interface can be considered as the action of transferring part of or complete UI from one computer to another one, such as from desktop computer to a handheld device [DSC*08]. Myers [Myer01] called DUI multi-machine user interfaces which spread computing functions and their related user interfaces across all computing and input/output devices available to a particular user or group of users. Lòpez-Espin et al. [LGL*11] define that the DUI is a user interface that some or all of its elements can be distributed across different displays or platforms and a DUI system is an application or a set of applications that makes use of DUIs. To characterize the DUI itself, several essential properties (or features for DUI systems) are posed. They are portability, decomposability, simultaneity and continuity. Another statement for DUI definition is made by Niklas Elmqvist [Elmq11]: a distributed user interface is a user interface whose components are distributed across one or more dimensions. There are altogether five dimensions and each dimension represents one way of how an application can distribute itself, i.e. input, output, space, time and platform.

Demeure et al. [DSC*08] introduced the 4C reference model to identify diverse DUIs. The model is a good complementary to the definitions of DUIs aforementioned and it characterizes DUIs explicitly with 4 dimensions: Computation, Communication, Coordination and Configuration. Balme et al. [BDB*04] proposed a reference model, in which a 3-layer conceptual architecture of the CAMELEON-RT reference model is presented as general strategies to the DUI solution. These reference models are developed only as conceptual solutions, whereas they lack of prototype implementation and evaluation. Many projects in the field of DUIs propose their architectures of solutions similar to this model. In the solution presented by Bandelloni et al. [BMP*07], there are service modules that just act as the platform manager and the adaptation manager. Manca and Paternò [MaPa11] as well as Nichols et al. [NMH*02] introduce their high-level platform independent UI abstractions to facilitate the adaptation of UIs across heterogeneous platforms using the similar architecture. Several UIML-based [Phan00] tools like MONA [BSS*05] have been published which focus on how the adaptation manager in this reference model can be built.

Our DUI approach is related to work in two research domains, namely mechanisms for distributing and migrating Web UI, and frameworks for using multiple personal computing devices to perform a single user task.

Distributing Web UIs means ungrouping Web document elements and presenting them separately without compromising application functionality. The granularity of UI splitting can range from arbitrary partitions to pre-defined UI blocks. Ghiani et al. [GPSa10] provide a mechanism to select a part of a Web page which can be migrated and shown on a mobile device. However, this approach is only feasible for the adaptation of Web pages and does not support presentation of different UI components on multiple devices at the same time. Model-based approaches [VVLCo5, BSS*05, LuCo05] define different abstract UI configurations at design time and generate concrete UI presentations at runtime. These works demonstrate dynamic distribution of Web interfaces among heterogeneous platforms.

But reusability and extensibility of sub-services/components are major shortcomings in these approaches. A new UI schema needs to be fixed for a complete application. Sub-service definitions cannot be separated. Consequently, the services of an application cannot be ported with ease. Learning to use the schema for an application induces additional development effort. Moreover, if a new application joins the system, new UI schema files must be written, and the root UI schema must be modified. In contrast, we consider Web applications composed of widgets using open Web standards.

Dynamic DUIs should support runtime component migration. Necessary steps for a successful migration are presented in the Roam project [CSW*04]. Roam preserves the application execution state information such as heap, stack, network sockets, etc. at the start of the migration and restores them after migration. For continuous Web browsing, Alapetite et al. [Alap10] migrate Web sessions across mobile devices using 2D-barcode captured by cameras. A dedicated State Mapper is also developed in [PSSc08] for state recovery during UI migration between mobile phones and digital TVs. Inspired by these approaches, our framework realizes complete continuous migration tailored to Web widgets.

Multi-device collaboration means that multiple devices can join the same application scope and that these devices can complete tasks together. Early approaches have focused on supporting desktop applications with devices such as PDAs and handheld computers over wired or wireless connections. Pebbles [Myer01] extends computing and I/O functionalities by involving heterogeneous devices. The extended UIs are native applications specially tailored for each computing platform and each functionality. Thus, multi-device UI are tightly coupled with the computing hardware. Melchior et al. [MGVV09] present a P2P framework that helps deploy distributed graphical user interfaces. All devices must install the framework before they can create components or import remote components directly from other devices. Many projects consider one-to-one mappings between users and devices, which is more applicable for collaborative scenarios. MarcoFlow [DST*10, DST*12] uses modular UI to represent the relevant controls and information to the user, but it focuses on the orchestration of business processes involving multiple users with different data views. Pierce and Nichols [PiNi08] use the idea of ownership to address personal computing devices and to enable seamless user experience over multiple devices. Their prototype simplifies the development of applications that are aware of a user's devices but it does not support UI migration. The DireWolf framework supports any device with an available modern Web browser. There is no need for pre-installed components or configurations. In the following, we first introduce Widget-based Web applications to clarify the context in which DireWolf was developed.

Widget-based Web Applications

Important prerequisites for distributing individual elements of complete Web applications are a clear separation into conceptual and functional units, a context for managing separation, and cross-device communication between these units. In this section we briefly introduce

widget-based Web applications and discuss why they fulfill the above prerequisites and thus served as foundation for the DireWolf framework.

The basic building block is a *widget*. Conceptually, a widget is a self-contained mini-application with limited, however clean-cut functionality. Widgets are usually designed to accomplish small stand-alone tasks, which may recur in multiple different applications. Furthermore, widgets are usually designed with limited display size, such that multiple widgets fit on one desktop browser screen or single widgets fit on limited-size mobile device screens. By design, widgets are reusable for multiple purposes in different applications. As such, widgets strongly resemble mobile applications. Technically speaking, existing widget standard specifications define widgets as packaged Web applications including means of configuration and access to dedicated widget application programming interfaces. Principally, any existing Web application can be “widgetized”. However, the form factor of limited display size often requires an adapted design. In practice, widgets usually serve as minimal frontends to more complex Web services. For our work, widgets perfectly serve as the functional units to be migrated across devices.

Complex applications can be achieved by orchestrating multiple widgets in a dashboard fashion in *widget containers*. Research towards the effective integration of widgets to complete collaborative Web applications resulted in additional layers on top of widget containers that make use of the DireWolf framework, i.e. *widget spaces* and *inter-widget communication*.

First, combinations of multiple widgets require a working context and technical support to manage such contexts. In our work, we employ the concept of a *widget space* [BSGi11] as working context. A widget space is a collaboration context, in which multiple users collaboratively manage and operate sets of widgets and additional resources to create custom applications for different purposes. For this work, we extended widget spaces by the additional notion of multiple devices per user.

Second, the integration of multiple widgets to complete applications requires an interoperable communication mechanism between widgets, referred to as *Inter-widget Communication* (IWC). With such a usually publish-subscribe-based mechanism, messages can be broadcasted from any widget and possibly dispatched by other widgets, thus allowing the orchestration [ZIBu11] and tighter integration of multiple widgets to complete applications. Most existing approaches only support local IWC, i.e. communication between widgets within one single browser instance. An additional feature of our complete IWC approach includes remote communication between widgets across different browser instances and users [GVD*11]. For this work, we use both forms of IWC as carrier for message exchange between different parts of our DUI framework within and across devices.

Figure 6.41(a) depicts the initial setting from which this work departed. In the following section we elaborate on the extensions contributed by our DUI framework in detail, thus leading to the situation in Figure 6.41(b).

DireWolf Framework

Based on the state-of-the-art in widget-based Web applications discussed in the previous section, we now introduce the DireWolf framework. First, we discuss the particular requirements for such a framework, which are not yet covered by existing widget-based Web application frameworks.

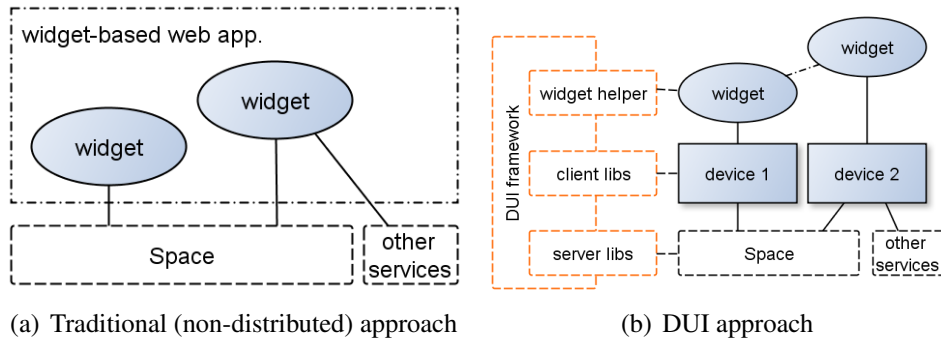


Figure 6.41: Common widget UI (a) versus a distributed widget UI approaches (b)

The DUI framework is involved in every layer of the widget-based Web application. As shown in Figure 6.41(b), components should be created for widgets, client browsers, backend services as well as the data storage. Framework client components are included in the widget application document rendered in the Web browser. They manage communication and synchronization between widgets on one device but also between widgets on other devices. The framework server components extend the functionality of common widget spaces with services for data persistence, user device profiles and shared application state. The DUI framework provides management services for device profiles and widgets when the user owns multiple devices. The inner workings of a widget are out of concern of the DUI framework. A requirement is that a mobile device needs to host some modern Web browser such as those found on most commodity smartphones and tablets. The use cases focus on creating, getter/setter and operating on resources (widgets).

Requirements Analysis of DUIs

As a first step, we performed a requirements analysis with the goal of improving deficiencies found in existing work on DUIs, thereby taking into account the current state-of-the-art in widget-based Web applications (cf. Section 6.3.2). Figure 6.42 provides a high-level overview of the main identified requirements for a DUI framework, grouped into four interrelated categories: *device information*, *device ownership*, *distribution & migration*, *application state* and *widget handling*.

A DUI framework must enable the management of general and context-specific device information. General information includes information on device *connectivity* and *profile*.

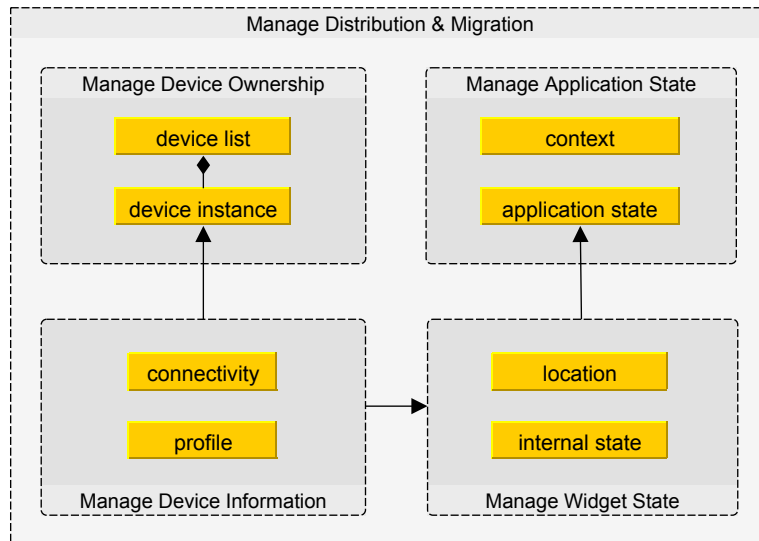


Figure 6.42: Requirements to a dynamic widget-based DUI framework

A device profile captures information on device type (e.g. smartphone, tablet, laptop) and capabilities (e.g. operating system, display size, in/output modalities, browser type) required for device recognition and adaptation purposes. Device connectivity describes the current availability of the device for collaboration and should be updated in real time. Context-specific information includes *device location*, i.e. in which context the device is currently active and *displayed widgets*, i.e. which widgets are displayed on the device in the current context.

Furthermore, a DUI framework must dynamically capture and manage *device ownership*. With the ever dropping prices of mobile devices, a person's device portfolio is likely to change often. Each user should thus be enabled to dynamically manage a personal *device list*. Thereby, each device instance describes a virtual device which can be bound to a real device. The introduction of virtual devices provides additional flexibility, i.e. multiple configurations for a single device and switching between real devices.

Obviously, a DUI framework must support distribution and migration of widgets across devices within a given context. In its simplest form, migration is a synchronized procedure controlled by the framework, where a widget is first removed from a source device and then created on a target device. However, constellations of widget distributions must be persistent. Thus, a DUI framework must be enabled to manage, store and synchronize application state within a given working context. For simple migration, application state must include information on the context and on widget locations, i.e. which widgets are currently residing on which device for which person. However, simple migration does not guarantee a seamless working experience. Although general widget configuration parameters are persistently managed by current standard widget engines, a widget will lose its internal state during the migration procedure. For some widgets this is not an issue (e.g.

a clock widget), for some it is. Thus, a DUI framework must support the management, storage and synchronization of internal widget state. With such measures, a DUI framework is enabled to support continuous migration, i.e. a widget stores a snapshot of its internal state before removal from a source device and restores internal state after its creation on the target device.

Framework Design

Figure 6.43 depicts the key architecture features of the DireWolf framework. As mentioned in Sec. 6.3.2, the DUI framework requires a real-time communication mechanism to “glue” all distributed UI components into one cohesive application. The *Message Router* server component provides bi-directional asynchronous message exchange between the client components and the server.

DUI Client is a widget helper component to be included as a JavaScript library in the widget namespace. DUI Client usage in widgets is optional (e.g. legacy widgets). These widgets can still be distributed and migrated. However, the DUI Client enhances DUI-related features for the widget and provides an API to interpret and create framework messages and events. DUI Client has additional methods to store widget state as part of application state at the server-side service component. It sends requests, and server components send back responses as well as broadcast notifications to all other Web clients if necessary.

DUI Manager is the central DUI component on the client browser. All features/functionalities are directly or indirectly related to it. DUI Manager connects to other components of the framework in three ways: request-response communication, local and remote IWC. For example, DUI Manager uses requests-response communication to retrieve user profile and space information from server-side services. Local IWC is used for communicating with widgets running in the same browser context. Remote IWC provides the message-exchange mechanism for widgets and DUI Managers located at different devices.

At start, the DUI manager fetches the user profile which contains the device list and the device profiles. The connectivity of a user’s devices is monitored constantly after the DUI manager is activated. The user can choose one virtual device per real device. If a device is not listed, the framework attempts to recognize it by using cookies, HTTP User-Agent headers and user input.

DUI Responder is the server-side central DUI component. All DUI relevant requests are redirected to this component. The main tasks of DUI Responder are to maintain DUI-relevant data and keep all DUI managers on client browsers synchronized.

Widget Migration

By using a widget approach, the dynamic transition of UI components from desktop to mobile devices is simpler. Widgets resemble mobile device screen sizes by design.

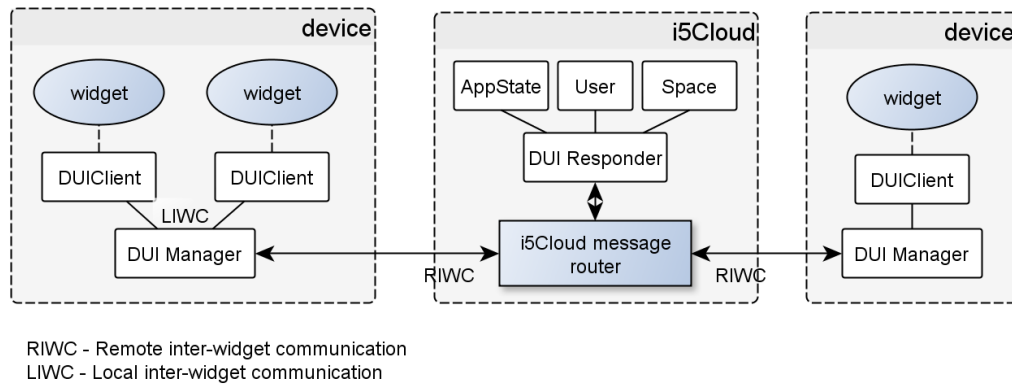


Figure 6.43: Abstract architecture of the DUI framework

Rendering a widget on smartphone or a tablet only requires adaptation of the widget containing element.

Considering the fail-over, since mobile devices can go offline unexpectedly, widgets can become inactive. The DUI Responder considers a widget to be inactive if it cannot find an active device displaying the widget. Different procedures are provided to inactive widgets and active widgets. Figure 6.44 illustrates the case of continuous migration. When a DUI Manager initiates a widget migration on any device, the DUI Responder looks for the widgets on all devices of the requesting user. If the widget is found to be inactive, the DUI Responder switches the widget location from no device or an inactive device to the migration target device. Then, it sends out a message to perform the migration procedure on all DUI Managers.

During continuous migrations, widget state is saved right before migration. The widget can retrieve state as a snapshot for continuing the task. DUI-supported widgets can be either inactive or active. DUI Manager tries to restore the state for inactive widgets and guarantees the continuity for active widgets. For inactive widgets, the steps are the same as the non-continuous migration of inactive widgets, except that DUI Manager sends the last saved state of the widget.

For continuous migration of active widgets, DUI Manager asks the widget's DUI Client to collect the widget state for the incoming migration. On receiving the command for migration, DUI Manager on the source device informs the DUI Client to prepare the widget removal. DUI Manager on the target device extracts information from the command. DUI Client is then guided by DUI Manager to run several steps to finish the migration.

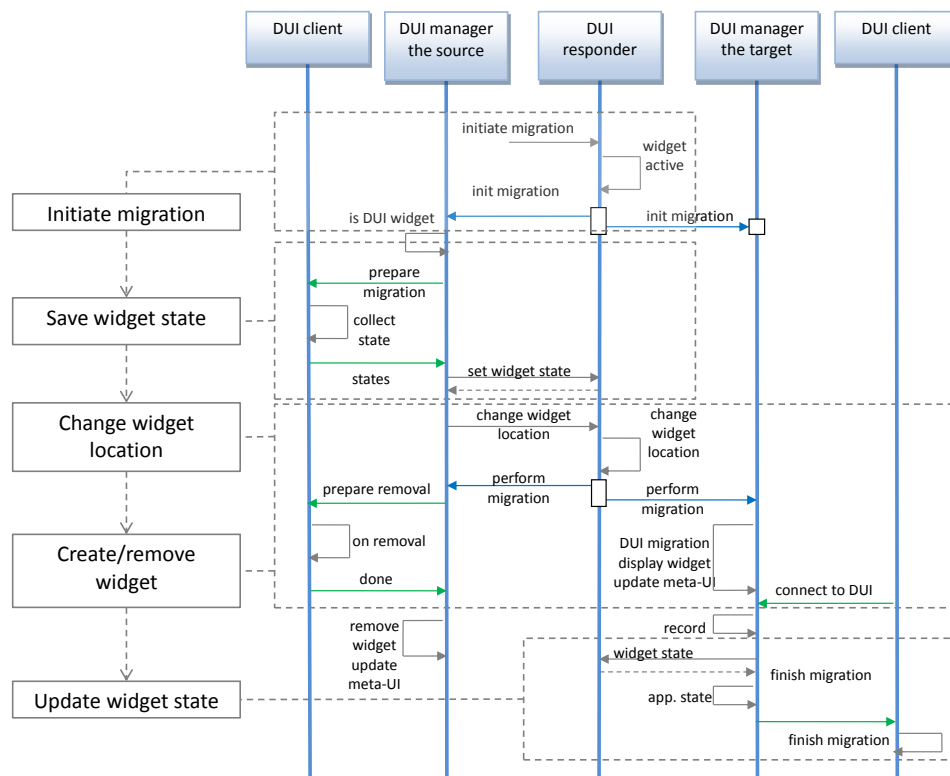


Figure 6.44: Sequence diagram of the continuous migration of active widgets

DireWolf Framework Implementation

The implementation of the DireWolf framework builds upon the Open Source Java-based ROLE SDK⁸ including a platform for hosting and managing Widget-based Web applications as described in Section 6.3.2. As basic widget engine, the ROLE platform employs the standard OpenSocial [OpSo12] container Apache Shindig⁹. On top of Shindig, the platform implements a set of RESTful services for *user management* and *personal and collaborative widget space management*. It should be noted that the space concept is currently standardized in the OpenSocial 3.0 specification. Consequently, it will be implemented in Shindig and will possibly become part of other Shindig-based widget platforms such as Apache Rave¹⁰. Furthermore, the platform supports secure authentication and authorization by employing OpenID and OAuth. A real-time service realizes the integration with a standard XMPP [Sain11] server providing support for multi-user chat conversations in widget spaces and publish-subscribe support for remote IWC. Associations between modules are realized by injection. For our work we strongly employ IWC, using HTML5 Web Messaging [Hick11a] for local IWC. An additional feature of our complete IWC approach includes remote communication between widgets across different browser instances and users [GVD*11] using the XMPP protocol [Sain11] and its publish-subscribe extension [MSMe10]. We use both forms of IWC as a carrier for message exchange between different parts of our DUI framework within and across devices.

On client side, the platform provides an AJAX browser frontend based on HTML/JavaScript/CSS and jQuery¹¹. For client-side real-time support the ROLE platform employs strophe.js¹², a robust XMPP library for JavaScript including support for XMPP over Web-Socket [Hick13] in modern browsers. Widget spaces are used as context for IWC. In collaboration with user and space management services, the platform real-time service manages one dedicated publish-subscribe channel per space for IWC including whitelist-based access control. On client side, every widget space is instrumented with a DUI Manager including an *IWC proxy*, which routes outgoing IWC messages to the affiliated XMPP server via the strophe-based XMPP connection and incoming messages to all widgets in the space via HTML5 Web Messaging [Hick11a]. Widgets can be equipped with IWC support by simply importing a small *IWC client* library and implementing functions for publishing and processing IWC messages. The DUI Client library extends the plain IWC library by a set of functions related to storage and retrieval of internal widget state.

Given that many technical prerequisites for DireWolf were already fulfilled by the ROLE platform, we chose an integration approach. In its current version, DireWolf is an extension of the existing ROLE platform and its components. The DUI Responder is realized as an additional RESTful service for managing device migration-specific data such as personal

⁸<http://sourceforge.net/projects/role-project/>

⁹<http://shindig.apache.org/>

¹⁰<http://rave.apache.org>

¹¹<http://jquery.com/>

¹²<http://strophe.im/strophejs>



Figure 6.45: DUI manager user interface in a widget space sidebar panel

device lists, device profiles, and user and space-related application states. Client side components such as DUI Manager and DUI Client communicate application state and initiate widget migration by simple HTTP requests to the DUI Responder, which in turn controls the synchronization process and initiates real-time synchronization necessary for migration. All migration-related communication between individual components (Message Router, DUI Manager, DUI Clients) is handled via ROLE IWC over a separate publish-subscribe channel to avoid interference with regular developer-defined IWC messages.

For convenient control of widget distribution and device registration DireWolf provides a set of user interface components as frontend to the DUI Manager. Figure 6.45 shows the main component integrated into the side panel of a widget space's view in the overall ROLE platform user interface. The upper *Device Manager* button bar provides shortcuts to a device manager console for personal device management including detailed configuration and debugging options. The *Current Device* resp. *Remote Devices* section lists all widgets displayed on the current device resp. remote devices along with device connectivity. In the example in Figure 6.45, the current widget space contains six widgets, distributed to four devices with different profiles (PC, iPad, iPhone and Mac). Only two devices are currently active, indicated by the green circle next to the device name. Thus, only five widgets are currently visible. One widget was previously migrated to the user's iPhone, which is currently disconnected, indicated by a grey marker. By using drag and drop, widgets can be (re-)distributed between active devices.

Evaluation

In this section, we present the setup of and results from our evaluation experiments which investigated the applicability and usability of the distribution of widget-based user interfaces in Web applications across different personal user devices. The evaluation of the DUI framework is divided into two parts. First, we measured and analyzed the technical properties of the widget migration operation. Second, we conducted an extensive user study for assessing the usability of the DireWolf framework. The evaluation targeted to measure the impact of the chosen technical framework upon the user experience, as well as to measure the user preferences and satisfaction, keeping into account the contrast to traditional Web applications. The methods used in the usability study try to discover the relation between different input and output possibilities and different devices, as well as the usage of more devices for achieving a customized personal computing environment, where users can interact with complex applications across multiple devices.

Widget Migration Performance

The migration component of the DireWolf framework was tested in a wireless local area network; a common environment at home or in office where a user would use DUIs. The ping latency of the network (6 ms) was considered negligible. Two setups were considered. The first setup measured migration between two desktop machines (Mac OS, Windows 7), using the Google Chrome browser (version 23). The second setup measured migration between desktop machines and an iPad 1 with iOS 5.0, using the Safari Web browser.

Tests were conducted with widgets with simple functionality, measuring the time between two consecutive migrations across two devices. In order to avoid noise induced by local time inconsistencies between devices, a reverse operation was automatically executed after initial migration, and total round-trip time was recorded. For consistency reasons, two kinds of migrations – simple migration (non-state-preserving) and continuous migration (state-preserving) – were evaluated. Round trip times for 100 migrations (i.e. 50 rounds) were measured.

Overall, our prototype achieved good performance results. For a blank widget, migration lasted on average $M = 0.36$ s ($SD = 0.05$ ms). Continuous migration requires two more steps than a simple migration, i.e. storing widget state and rendering the widget with the Apache Shindig rendering engine. Average time for continuous migration between the MacBook and the desktop computer was $M = 1.31$ s ($SD = 0.15$ ms). Due to the hardware differences, MacBook and iPad combination yielded higher average migration time ($M = 2.06$ ms, $SD = 0.22$ ms).

By decomposing the time necessary for the migration and observing the interval needed by each component of our framework, the results show that the initiation and the widget rendering process take more time than the migration itself. The Shindig server's JavaScript library loading and the widget rendering steps require approximatively 69% of the time. In contrast, the loading time needed by the DUI components is less than 25% of the overall time.

Usability Analysis

In this section we report on two user studies conducted in which the participants performed the tasks distribution of user interfaces as well as widget transitions across various devices. The 25 participants were students or young researchers, studying in different domains at university level. At first, participants were familiarized with the DireWolf framework, the concept of Web widgets and spaces. Next, the multi-device distribution of user interfaces was demonstrated on a simple widget-based Web application. Finally, the users were asked to perform tasks in two subsequent experiments. The first one tested the user preferences for performing certain activities using various widget types and devices. The second studied the user performance and the user experience with a complex Web application for video annotation with and without distribution of the user interface. The user studies were arranged in individual sessions, after which each user had to complete a usability questionnaire. The answers of the participants were collected and compared. In addition, user interactions with the prototype were measured.

Widget Preferences

In this experiment we assumed that users can perform certain tasks like painting or typing better or worse, depending on which type of device like touch-screen or desktop computer they are working on. Three different widgets were involved in this experiment (cf. Figure 6.46): a *painting widget*, a *text input widget* and a *map widget*. The participants were asked to draw a given picture on a canvas using both a touch-screen device and a desktop PC. The drawing performance in terms of time needed to finish the drawing was measured. This type of widget was used because the painting action requires accurate targeting and continuous movements on the screen. In the text input widget case, the participants were asked to type a given text. The times for finishing text input using on-screen vs. physical keyboards were recorded. This widget assesses how well the on-screen mimicks the physical keyboard and how people react on these two input modalities. In the map widget case, the participants were told a set of local sights equally known by all persons. The time for locating these places on a map was recorded. This task required a different set of interactions depending on the device used. For the touch-screen device, multi-touch gestures were involved like swiping and two-finger zoom. For the mouse supported device, actions such as wheel zooming and dragging were involved. The map widget assessed how people react on these two sets of interactive operations. Upon task completion, participants stated they preferred device/widget combinations for different tasks.

Although users perform better on touch screens for drawing (mouse supported device: $M = 19.4$ s, $SD = 5.4$ s; touch screen device: $M = 15.1$ s, $SD = 4.2$ s), 67% of users preferred the mouse device. The typing task revealed that 96% of users preferred hardware keyboard, indicating that the touch screen keyboard only provides an alternative, but not an optimal solution (performance results with hardware keyboard: $M = 48.7$ s, $SD = 6.7$ s; touch screen keyboard: $M = 108.7$ s, $SD = 23.6$ s). The map navigation task showed that although people perform better on mouse supported devices (mouse supported device: $M = 39$ s, $SD = 9.4$ s; touch screen $M = 48.9$ s, $SD = 10.3$ s), 56% of them prefer to use the touch screen for

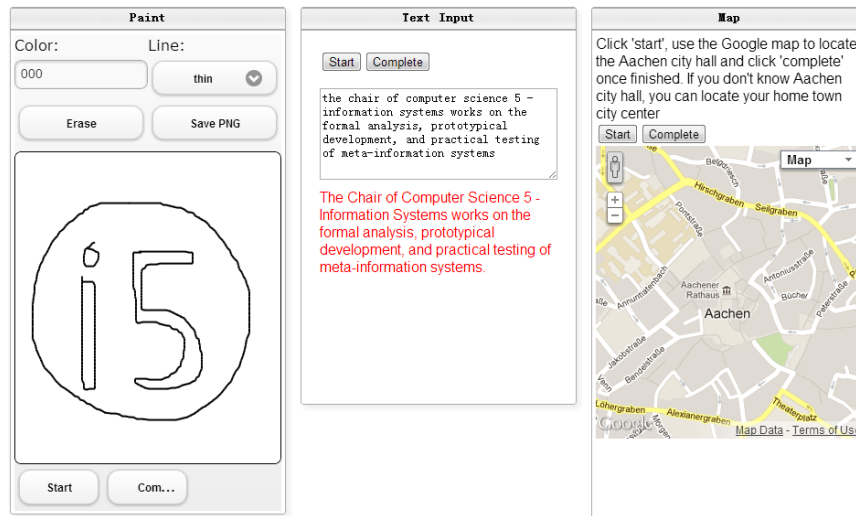


Figure 6.46: Three different widgets used in the DireWolf user preference evaluation

the interaction. This result shows that touch screens already exceed traditional desktop computer capabilities for this particular type of interaction.

The results obtained after the evaluation show that both the task completion time and the user preferences for the usage of the three widget types. It can be observed that although users perform better on touch screens for drawing, 67% users prefer the mouse device. This is mostly due to the novelty of the action, as most users declared that they have never painted on touch screens before. Concerning the typing activity, 96% of users prefer hardware keyboard. The fact indicates that the touch screen keyboard only provides an alternative, but not an optimal solution. Finally, regarding the map navigation, although people perform better on mouse supported device, 56% of them prefer to use the touch screen for the interaction. Considering that the touch screen devices came out much later than the desktop systems, this result shows that the touch screen already exceeds the traditional desktop computer capabilities for this particular type of interaction.

The evaluation results of this study support the initial assumptions about device preferences for different tasks. This implies that there is a need of widget distribution and migration. Overall, the preferences do not only depend on the performance, but also on the interaction style and the familiarity of the users. For the paint widget most users declared that they had never painted on touch screens before. The results obtained using the paint and the map widgets exemplify the mutual influence between the familiarity and style of interaction. Furthermore, the result of the text input is an example for the overwhelming influence of the performance factor.

DUIs for complex Web applications

This part summarizes the second usability study, which evaluated the usefulness of the DireWolf framework for distributing the interface of a *complex Web application*. We validated the benefits of widget-based DUIs across devices, while the user performs a

complex UI interaction task. This experiment used SeViAnno 2.0 (cf. Figure 6.40), our widget-based research prototype with a customizable and distributable user interface for semantic video annotations [CRJ*10]. The application is composed of five widgets, fulfilling the following functionalities: video player, adding annotations, listing of annotations for display and navigation, annotating places using a map and listing all the available videos. A widget can migrate between devices without losing its current state. Thus, after distributing the SeViAnno 2.0 widgets to multiple devices, the widgets continue their operation while the whole application preserves its integrity and functionality. For example, after migrating the video player widget it resumes playing without any interruptions.

In our study, SeViAnno 2.0 used a desktop PC to display a full screen video player, an iPad to enable video navigation using the list of existing annotations and an iPhone to display the map widget for spatial annotations. Users were asked to watch a video clip and locate the time point of a certain event. Using this setting, we demonstrated the usefulness of distributed interfaces by showing that content can be found faster. Two ways in which a user could locate the right time point in a given time frame were considered. One way was to randomly browse through the video using the video seek bar. The second way was to search information in the video annotations list, where content about persons, events, places, objects, etc. was available. In the evaluation scenario, the first search method is used when the user does not distribute the widget and watches the video clip in full screen mode. With a full screen display, the second search method is available only if the widget is located on another device.

Considering a single video, the first task was to locate the occurrence of a known person in the video and the second task to locate the background voice that introduces a person in the video. The time spent for each of the two search tasks is depicted in Figure 6.47(a). The data recorded is the time that the user spends on locating the right time point in the video.

After the experiments, users were free to rearrange the widget distribution and to perform other tasks. One of the observations at this stage was that due to user preference on single widgets and the multi-platform compatibility of the widget implementation, different users can have different preferences on the distribution of the widget set (see Figure 6.47(b)).

The results show that the users got the information faster with the help of the annotations widget. Considering the deviation, the search was more reliable with the help of the list of annotations. In the results, a high deviation implies that the task is done by chance. This became more evident as the information was harder to locate. Under the condition that the users watched the video in full screen mode, such results imply that the distribution among multiple devices improves the usability of complex Web applications.

Figure 6.47(b) shows the preferences on how users distributed the SeViAnno widgets. As it can be observed, the users preferred to place the interactive widgets on the tablet. Users chose the desktop device for widgets involving text input, as observed in the previous experiments. The mobile phone was less used by users, because of the small screen size. This also implies that users do distribute the widgets based on their preferences.

Finally, we have also gathered general user feedback, using a section at the end of our

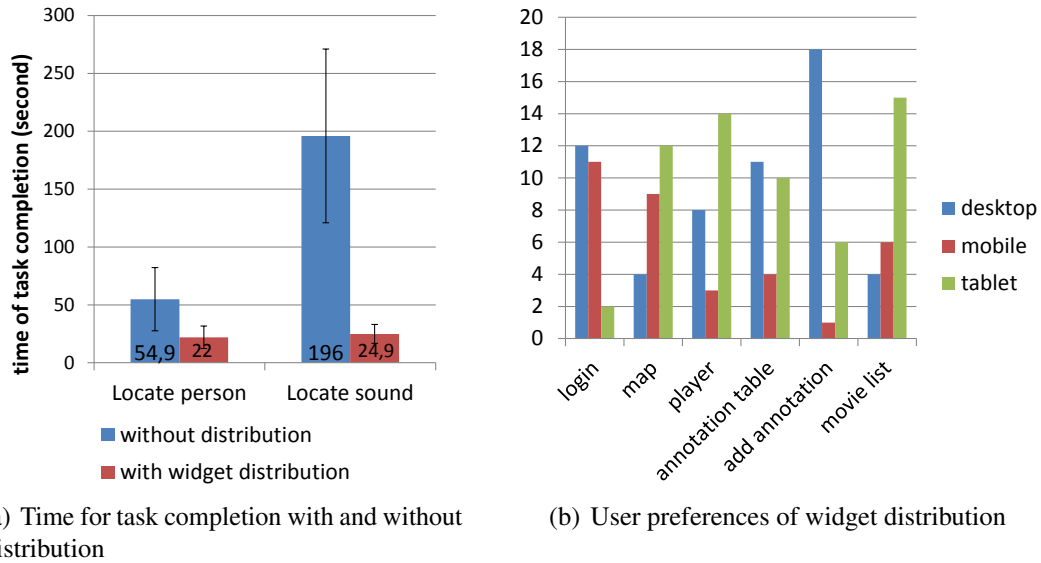


Figure 6.47: Results of the evaluation using SeViAnno 2.0

questionnaire for general comments. For the implementation of the complex application, users expected better widget optimization for mobile devices. Related to the DUI framework, users expected a reduced amount of side panel elements (e.g. information about widgets, users) when there are many widgets in the space. Furthermore, different interaction capabilities for the widget migration have been mentioned, such as drag-and-drop of the widget itself to the target device. We consider these comments for further prototype improvements.

The presented evaluation is limited to technical properties of the widget migration feature. However, we conducted an extensive user study for assessing the usability of the DireWolf framework, which due to space limitations could not be discussed in this paper. In addition, DireWolf is currently being tested on a bigger range of devices.

6.4 Summary

This chapter presented practical results and observations obtained through the realization of the CAELUS architecture. Several prototypes in the form of advanced services and information systems were implemented which are distinguished in three groups here, i.e. cloud platform, mobile multimedia and personal cloud computing. These prototypes have been applied in the domains of cultural heritage management, technology-enhanced learning and human-computer interaction. They were successfully designed, realized and deployed; thus, demonstrated the applicability and validated the soundness of the CAELUS concepts.

This section summarizes the research prototypes and their evaluations under the big picture of the CAELUS architecture. In the second part, the combined view of concepts and

approaches is assessed through the main three facets proposed in Section 4.1. In connection with this summary, Section 7.2 from the next chapter extends the results into an outlook for future work.

6.4.1 Discussion

A combination of different approaches and algorithms can play an important role in delivering fast and intelligent video processing services for a better mobile UX. State-of-the-art mobile video UX enhancement techniques were combined with the cloud computing paradigm in our prototype called MVCS. Video stream navigation on mobile devices is eased by segment cues and tags automatically generated by intelligent processing means. Additionally, the cloud services adapt the zooming level of the video streams to overcome the problems with small screen sizes. The evaluation revealed that the utilization of a cloud environment for a parallel processing of video chunks enables near-real-time delivery of complex tasks. In the light of these findings, we are convinced that improvements of user experience in sharing mobile video applications have been achieved. To overcome the limitations of mobile phones, solutions like finding optimal zooming ratio [KPSV07], region of interest enhancement [STWD10], and segment-based video browsing [BZPr10] are used. We embrace these approaches and we go a step further to transform them into reusable cloud services.

MACS middleware showed that the local execution time can be reduced a lot through offloading, which is sometimes not acceptable for users to wait for, and pushing the computation to the remote cloud can lower the CPU load on mobile devices significantly. Meanwhile, energy can be saved which indicates users can have more battery time compared to the local execution. The results also prove that the overhead of our middleware is small. Previous work [KAH*11, SBCD09, ChMa09] has proposed many mechanisms that address the challenges of seamless offloaded execution from a device to a computational infrastructure (cloud). MACS offloading distinguishes itself by extra profiling and resource monitoring of applications and adapts the partitioning decision at runtime. Moreover, we consider that application developers can better organize their application logic using the established Android service design patterns and benefit from the MACS middleware.

XMMC services, and the AnViAnno and SeViAnno applications facilitate ubiquitous multimedia content and metadata management. Since a lot of scenarios for mobile collaboration demand real-time functionality, more and more professional communities adopt such services. Cultural heritage management is a practice we accompanied since many years [KSJ*05a]. Here, we presented a collaborative semantic-enhanced multimedia annotation system. We developed collaborative annotation and editing services and integrated them with existing mobile augmented-reality browsers and existing multimedia services based on the MPEG-7 standard. Real-time support was realized by deploying the Extensible Messaging and Presence Protocol (XMPP). These services have been used in a cultural heritage documentation scenario which exposed many of the addressed requirements. Evaluation

indicates that such solutions increase the awareness of community members for activities of co-workers and the productivity in the field in general. Many systems are available to store, consume and annotate multimedia. YouTube is an example of the prosumer paradigm expansion, where users can share their own videos in certain communities or open on the Web. However, there is limited support for peer production in on-site communities of practice by means of multimedia annotations combined with MAR, through real-time collaboration. With CAELUS we provide an infrastructure for real-time collaboration for annotating multimedia, available for both mobile applications and Web applications. Other related works [LuMa09, WMLa10, Gerl07, SSSc10] present services for collaboration but lack to support multimedia metadata. To achieve interoperability and interchange between such shared knowledge based, the metadata has to be based on standards. We, therefore, use the (XML-based) MPEG-7 standard for managing the semantic annotations.

The Learn-as-you-go micro-learning prototype leverages the availability of i5Cloud services to support a spatio-temporally-distributed learning workflow. It realizes seamless learning experience on smartphones and the Web 2.0. The cloud approach with advanced data synchronization technologies lowers the barriers in cross-platform learning processes and knowledge management using Web 2.0 multimedia artifacts. Mobile tools such as CAMCLL [AHZh09] and MicroMandarin [ESC*11] have applied the micro-learning concept for language learning on mobile phones as well. However, they are limited by the services they include. Our tools allow the user to use any Web resource or service and to grab the relevant information. The existing (semi-)automatized tools [LGZh03] for grabbing data from Web sites are focused more on extracting structured data from e-commerce sites, therefore, their usability for learning purposes is limited. Our tools allow users to grab any fragments from Web pages that are best contribute to his/her understanding.

With DireWolf, we try to leverage the lack of dynamic interactive environments based on Web technologies which can take advantage of the various personal devices used by an individual. We provide a framework that can facilitate user interactions on a federation of personal computing devices, by making use of distributed user interfaces. Our framework also provides features for distributing and migrating widgets, at the same time hiding the complexity of device awareness, communication and session mobility. As initial evaluation indicates, the framework adds only small overhead to the overall widget rendering process. DUIs have become a new topic in recent years and it has exceeded the traditional understanding of user interfaces. Model-based approaches [V VLC05, BSS*05, LuCo05] define different abstract UI configurations at design time and generate concrete UI presentations at runtime. These works demonstrate dynamic distribution of Web interfaces among heterogeneous platforms. But reusability and extensibility of sub-services/components are major shortcomings in these approaches. A new UI schema needs to be fixed for a complete application. Sub-service definitions cannot be separated. In contrast, we consider Web applications composed of widgets using open Web standards. Pebbles [Myer01] extends computing and I/O functionalities by involving heterogeneous devices. The extended UIs are native applications specially tailored for each computing platform and each functionality. Thus, multi-device UI are tightly coupled with the computing hardware. Our DireWolf

framework supports any device with an available modern Web browser. There is no need for pre-installed components or configurations. Most of related projects also lack of user management and support for DUI device management.

On a higher level, the CAELUS-based prototypes presented here aim to answer the second and third research questions from my dissertation. MACS middleware and DireWolf framework are examples of novel application models that operate on the intersection of cloud and mobile integration. They provide development support in terms of guidelines, reusable libraries and abstracted mechanisms for distributed computing. The cloud-augmentation reference model describes the cloud and mobile integration. Empirical results demonstrate performance gains and usability improvements. Furthermore, advanced mobile multimedia information systems can be realized by using the CAELUS architecture and i5Cloud platform. Collaborative and ubiquitous multimedia services such as XMMC support professional communities with both their work practices and a lot of new emerging practices. For example, the fusion of Web and mobile technologies allows professional communities to transform their collaborative work practices in the field. Moreover, the overall mobile user experience with mobile multimedia can benefit from cloud-based services as shown with the MVCS.

6.4.2 Achievement of Facets

This section takes the general facets described in Section 4.1 as a projection framework for the realization of CAELUS concepts and the dissertation objective. Figure 6.48 shows a raw mapping of the previously described CAELUS prototypes to the general facets. One can observe from the mapping that the main functionality of each prototype spans over two facets. However, the prototypes exhibit functionalities and features that fall into the remaining perspective to certain extent, but they are not depicted for the sake of simplicity and highlighting.

System and Technology Facet

Technology advancements within the last decade created the basis for emergence of cloud computing. On the other hand, cloud computing's vision of utility-like pay-as-you-go strategy drives innovation on all layers of compute, storage and networking technology stacks. This dissertation aimed at investigating on applying new technologies and identifying innovation potentials in mobile multimedia clouds. The prototypes and services presented in this and the previous chapter contributed to each sub-perspective from the system facet. The DireWolf framework and Learn-as-you-go implement data management services for personal clouds that span across multiple devices. The main contribution for this sub-perspective is the synchronized data availability. Moreover, MACS middleware implements a basic caching of application data and executable binary code between the device and cloud. The communication sub-perspective has been addressed through the realization of

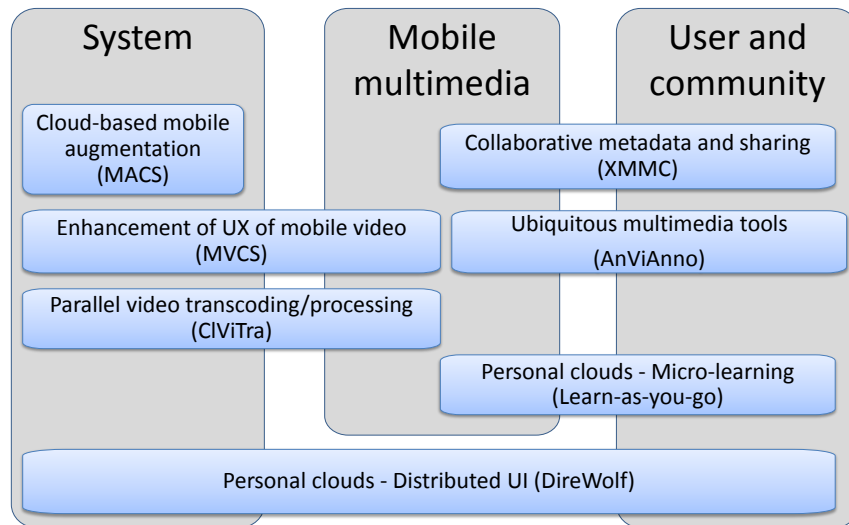


Figure 6.48: Mapping of CAELUS-based prototypes to the respective facets

real-time video streams sharing in MVCS and the realization of near real-time metadata messaging over the XMPP protocol. In general, i5Cloud platform services contribute from the computation sub-perspective, which is understandable considering the resource demands of media operations. CIViTra realizes video format transformation to adapt content for various clients. MVCS goes a step further by adapting video streams based on content parameters and user preferences. Both types of transformations adopt cloud-based computation distributed over a set of commodity machines. Moreover, i5Cloud achieves compute interoperability between different infrastructure providers.

Mobile Multimedia Facet

Rich multimedia applications on mobile platforms can benefit from cloud computing and storage capacities to finally enable a new generation of ubiquitous multimedia experience. Mobile multimedia is the central artifact for all services realized within the scope of this dissertation. The main contributions within this facet are brought from the content adaptation sub-perspective. CIViTra and MVCS realize services for content adaptation and transformation. The adaptation is guided by user and device factors. Moreover, MACS has demonstrated video enrichment by using the cloud-based augmentation middleware. The multimedia modeling sub-perspective has been partially realized through XMMC and AnViAnno through the LAS-compatible services for MPEG-7 based annotation of mobile multimedia artifacts. The multimedia semantics sub-perspective has been explored via community-based tagging in the micro-learning prototypes.

User and Community Facet

All the services and information systems presented here exist to enable users and communities to achieve their intended tasks. Therefore, many user evaluations were performed to assess the benefits of the CAELUS approach. MVCS aims to solve discrepancies with mobile video sharing, i.e. browsing and navigation of video streams. XMMC envisioned and realized a collaboration workflow with rich mobile multimedia such as augmented reality for digital documentation in the cultural heritage domain. DireWolf, on the other hand, demonstrated a collaboration on a single user level but between different personal computing devices. Applications such as AnViAnno and Learn-as-you go delivered solutions for ubiquitous multimedia sub-perspective on heterogeneous devices, via both mobile and Web based clients. The security and privacy sub-perspective has received the least attention in this dissertation. Only the MACS offloading approach can be enlisted as a contribution in this direction, by providing a natural mechanisms for privacy preserving.

After all is said and done, a lot
more will be said than done.

Aesop (c. 620 - 564 BC)

Chapter 7

Conclusions and Future Work

In this dissertation, research into mobile multimedia, and in particular, the overlapping area with cloud computing is presented. The dissertation has touched on many subjects, such as modeling and prototyping multimedia cloud platforms as a service, ubiquitous services, mobile user experience, mobile learning, human-computer interaction, development support, and evaluation. This chapter concludes the dissertation with a summary of results and contributions as well as outlining potential directions for future research work.

7.1 Summary of Results and Contributions

The overall goal of this work was to investigate possible modalities for applying the cloud computing paradigm to mobile multimedia services and applications. During my research, cloud computing was relatively well explored with regards to enterprise information systems, but was poorly applied for mobile multimedia systems. Cloud computing has transformed the practices of how services are designed, realized, deployed and provisioned, and ultimately how businesses are operated. Features such as scalability, availability, performance, fast time to market entry, and offline-resilience can be achieved, if the integration of cloud computing with mobile multimedia is realized. In addition, the same transformations can also influence mobile multimedia services as well. However, deficits in the engineering processes exist, despite the ever increasing popularity of mobile production and consumption of digital content in various amateur and professional domains.

This work has shown that cloud computing has an impact on every stage of the multimedia life cycle, and consequently, on multimedia services. Firstly, mobile multimedia sharing and collaboration benefit greatly from the cloud availability and geographical distribution. Secondly, multimedia is more complex than standard enterprise applications, i.e. it requires more resources per user session. Therefore, the scalability of the resources per user basis is important in order to serve as many sessions as possible. These and many other challenges for the engineering of mobile multimedia cloud systems were characterized through three main facets in Section 4.1. For each facet and sub-perspective, a set of opportunities

were named which could help the design of large-scale mobile multimedia applications. They served as a stepping stone in the derivation of a set of key requirements specific to this dissertation. The requirements were translated into the CAELUS architecture and methodology (see Section 5.1) which integrated cloud principles into the design and provisioning of mobile multimedia systems. It was also argued, that we have to try out and evaluate new standard and protocol suites combined with the mobile cloud computing delivery model in order to improve current development practices and to shape mobile and Web multimedia convergence. For instance, HTML5 [HTML5] has great potential to remedy the mobile device fragmentation with XMPP gradually taking the role as a cloud protocol [HoWa10, KCK10, SStr09]. The experimental evaluation with cloud video transcoding (see Section 6.1.1) and multimedia content-based retargeting (see Section 6.1.2) proved the achievement of cloud characteristics (such as scalability, multi-tenancy, service-oriented approach, auto-scalability and capacities for large volumes of data and big computation workloads) with the CAELUS architecture and i5Cloud platform. Several other cloud features (such as pay-per-use utility model, self-healing and service-level agreements) were not realized mainly because they are out of the scope of this dissertation.

Furthermore, three major reference models for the realization of cloud and mobile integration were identified. The “cloudified” server model resembles the traditional and most widely applied approach of provisioning services as a client/server architecture (see Section 4.2.1). Several considerations regarding the migration of services from a simple server to a “cloudified” server architecture were also distinguished. On the other hand, the cloud-based augmentation model proposed a lightweight application partitioning and a mechanism for seamless adaptive computation offloading. This novel mobile computing model places the mobile device in the center of control where the cloud serves only as generic infrastructure which is opportunistically accessed by the device. Between these two extremes, the fog computing model was identified as a balance when neither of them were deemed suitable. The trade-offs of each model are summarized in Figure 4.6 which show that there is no universal solution in the design of a complete and comprehensive set of mobile multimedia services. A service provider can then choose what kind of model best suits the application needs (see Table 4.2).

The impact of the cloud paradigm in the CAELUS architecture propagate beyond the infrastructure and platform levels. CAELUS and i5Cloud formed the basis for practical realization of advanced mobile multimedia services and complete information systems. Multimedia semantics and real-time collaboration are essential services for many mobile multimedia information systems as the workplace gradually becomes more on-site oriented. The AnViAnno and SeViAnno tools implement mobile and Web multimedia convergence (see 6.2.1) while XMMC realized real-time collaboration over multimedia content and metadata 6.2.2. XMMC showed how CAELUS-based services can transform a mobile AR browser into a collaborative AR tool for both supporting work of professionals in cultural heritage and increasing awareness for amateurs in a cultural heritage domain. In addition, cloud-based data synchronization combined with other multimedia services were the engine of mobile micro-learning system demonstrating the applicability of the CAELUS approach

in the technology-enhanced learning domain. CAELUS prototypes were also validated in the human-computer interaction domain where the MVCS services demonstrated the applicability of i5Cloud for improvement of video stream sharing. DireWolf envisioned scenarios where a single person used multiple commodity personal computing devices in a federated manner for achieving a single user task such as complex video semantic annotation. In summary, several CAELUS-based prototypes were successfully designed, realized and deployed in different domains.

Alternatively, the contributions can be summarized through the three faceted view described in Chapter 4. The three facets served as general framework to the analysis of the most relevant and complementary requirements, in order to achieve the dissertation objective and answer the proposed research questions. In general, the three facets provided complementary views of the same objective. As a result, the work in this dissertation tried to balance aspects from the three facets. To begin with, in the scope of the system and technology facet, CAELUS was realized using state-of-the-art cloud technology. Open-source projects with strong community commitments such as Hadoop and OpenFire XMPP were preferred. In addition, test bed and prototype systems and services were deployed and tested on our local and public cloud infrastructure (Amazon WS). The strongest contribution from system facet perspective was in the multimedia computation, realized as distributed processing and transformation of multimedia files. Data management was tackled at both basic content storage level and advanced metadata management level. Real-time and near real-time solutions for multimedia content and metadata streaming brought insights from the communication sub-perspective. Furthermore, looking from the mobile multimedia facet, this dissertation engaged with various multimedia types such as videos, images, augmented reality, and appropriate metadata. Content adaptation accounts for a substantial part in several cases within the dissertation. On the multimedia semantics and modeling levels, existing approaches from our previous projects such as Virtual Campfire were reused. In addition, new cloud-based services for semantics-based enhancement of user experience with mobile multimedia were deployed. Finally, user and community facet played a major role in the whole research process. We followed an iterative user-centered design process that places focus on mobile collaboration with multimedia artifacts and enhanced user experience using mobile services and distributed user interfaces. Heterogeneous devices were used in mobile settings, i.e. ubiquitousness was emphasized and tested. In summary, the overall work presented in this document successfully fulfilled different aspects of the three facets.

The concepts presented here were realized within the German excellence cluster UMIC (Ultra Mobile High-Speed Information and Communication) funded by the national science foundation of Germany. A major part of my research was also supported by the B-IT Research School. To foster the exchange of knowledge and experience in the mobile cloud computing community, we were involved in the organization of two international workshops, i.e. the Workshop in Future Mobile Applications and Services (FuMAS 2010) and the IEEE PerCom Workshop on Pervasive Community and Service Clouds (PerCoSC 2011). In the context of this work, several publications at international conferences, workshops and

journals were conducted which can be classified as:

Research question 1 - Multimedia cloud platform

- Scalable multimedia cloud architecture [KCK112, KoK110, KRKC10]
- Cloud-based video transcoding and processing [KCK111b]
- Enhancement of user experience with mobile video streams [KCK113]

Research question 2 - Innovative cloud-based application models

- Cloud-based computation and storage offloading [KYK112]
- Dynamic code partitioning [KoK112a]
- Elastic mobile applications [KCK111a, KoK112b, KCK110]

Research question 3 - Advanced mobile multimedia services and information systems (re-)defined by the cloud paradigm

- Ubiquitous multimedia and metadata collaboration [KNK113, KAK112, KoK109]
- Personal cloud computing [KRNK13, KCKJ11, BrKo13]

7.2 Future Work

Typically, after each prototype developed, each system evaluated, and each research paper published, a number of new issues that open new research directions appear. The emergence of cloud computing has made a big impact on the entire life cycle of mobile multimedia. Cloud systems exhibit features of complex information systems. Thus, further steps need to be done in order to fully realize the vision of a mobile multimedia cloud. Various potential extensions to the presented CAELUS architecture and methodology are possible. The following paragraphs present possible future research directions for mobile multimedia cloud computing originating from the research done this dissertation. Here again the three faceted view is used to group the research directions.

From a system and technology point of view, multiple new advancements can be explored. For example, we are developing more multimedia services that benefit from i5Cloud. Currently, the multimedia metadata employed is limited to only one standard, i.e. MPEG-7 Semantic Base Types. In order to have integration with other video platforms, and more importantly, to incorporate annotations from other sources, mapping tools are required. The W3C Media Annotations Working Group¹ works on a promising solution to providing an ontology and API for media object cross-platform integration. Additionally, there are open issues in i5Cloud such as more optimized resource scheduling where only an experimental

¹<http://www.w3.org/2008/WebVideo/Annotations/>

validation can incorporate the complexities of such architectures and infrastructures with specific features [GJQu09]. There, i5Cloud is being upgraded to use the OpenStack open source project since it is gradually becoming a de facto standard for cloud interoperability and is supported by all major cloud vendors. A lot more research opportunities lie in applying state-of-the-art computer vision and machine learning algorithms to the cloud platform. Until now, only small subsets of algorithms have been successfully ported to benefit from the cloud approach.

In the cloud-based augmentation direction, the next steps are to enable parallelization of the offloaded services. It means that instead of executing offloaded services again on single processor architecture in the cloud, the offloaded services can benefit even more from hardware accelerations. However, this complicates the offloading integrity checking and the development process. Additionally, we can extend the current middleware so that it supports automatic partitioning of arbitrary mobile applications. A greater challenge is figuring out how to estimate the characteristics of an application depending on different input parameters, which is precisely the relationship between the input of the invoked method and the execution time. Works such as [LROf08, YKSG94, WEE*08, MLP*06] serve as a starting point for estimating the worst-case executing time for certain parameters via combining compile-time analysis with profiling. This could help in MACS to characterize the relationship between execution time and input parameters by running the target application several times and adapting the offloading algorithm.

In a wider system perspective, mobile multimedia services can benefit from larger developer support efforts. For example, abstracted solutions for arbitrary relational databases which automatically perform the heavy-lifting of collaboration, sharing, synchronization and versioning would drastically change the development process for multi-device multi-user applications. However, this is not a trivial problem since mobile devices often operate in disconnected and decentralized modes. Creating services that supports these modes of operation would require large development efforts. Moreover, the Web 2.0 development process entails the ability to quickly deploy new features and scale with the number of users. Instead of having consistent and complete answers to each database queries, certain services would benefit from declarative consistency requirements which can be specific to the application. For example, the query language can provide guarantees for query performance at the cost of correctness and consistency.

As mobile device technology continues to improve, the scope of mobile multimedia facet increasingly extends with opportunities for fruitful future work. In adapting multimedia content for mobile services, future work could entail improvements in order to enable personalized video streams, overlays with additional information layers, interactive zooming and ROI. Further work needs to be done in order to establish whether the proposed i5Cloud approach leverages fast prototyping of mobile video applications from a developers perspective. Furthermore, using scalable video coding techniques we could reduce the footprint of the adapted content. The new DASH standard is a promising solution for mobile streaming to heterogeneous devices and a widespread Web integration. A GPU-based cloud

infrastructure would accelerate video processing even more.

The user and community facet also encompasses directions for improving mobile multimedia services. In the personal cloud computing direction, we implemented the micro-learn model as a proof of concept which can be extended to each learning phase. When planning, knowledge capture should be improved to more intelligent information scrapping on the Web, and enhanced context and activity capturing with OCR improvements on the mobile. At the learning phase, XMMC services could be employed to enhance collaborative learning in the cloud among learner communities. During reflection, context-aware learning content organization and reuse can be developed. In addition, the DireWolf framework paves the way for many interesting experiments. We envision our framework to be in the domains of technology-enhanced learning and interactive smart television. We also consider using the emerging WebRTC project for real-time browser-to-browser communication without an intermediate server. As a next step beyond the personal multi-device distributed computing environment, we will extend DireWolf to support multi-device multi-user collaboration. Further research must address security and privacy issues in message exchanges across devices and users.

In the UMIC cluster, future scenarios for high-demanding bandwidth usage are created to challenge current constraints on mobile phones such as battery capacities, screen sizes, input handling etc. Therefore, the evaluation of some prototypes was also considering many of these constraints. Furthermore, parts of the research (i5Cloud, micro-learning, DireWolf, AnViAnno) are further extended in the context of the Learning Layers (Scaling Up Technologies for Informal Learning in SME Clusters) project, funded by the European Commission under the seventh Framework Programme. The various collaboration methods mentioned here and the corresponding background relating to the semantic annotation scenarios can serve to the scaffolding of informal learning in heterogeneous networks.

In my personal opinion, mobile cloud computing is an intermediate stadium to the materialization of Mark Weiser's vision from more than 20 years ago in his article "The Computer for the 21st Century" in Scientific American magazine [Weis91]: *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it. [...] then we are freed to use them without thinking and focus on other goals."* Mobile cloud computing is a step towards a technology that is pervasive yet invisible. It is undoubtedly a powerful paradigm shift that is going to deliver advances in technology. Moreover, cloud computing will be key in the transformation of everyday life and work practices. Developments in cloud technology and mobile multimedia have changed how people are interacting with one another. It will continue to have impact on knowledge society by creating new markets where the geography of the workplaces will have diminishing importance.

Bibliography

- [AFG*09] ARMBRUST, MICHAEL, ARMANDO FOX, REAN GRIFFITH, ANTHONY D. JOSEPH, RANDY H. KATZ, ANDREW KONWINSKI, GUNHO LEE, DAVID A. PATTERSON, ARIEL RABKIN, ION STOICA and MATEI ZAHARIA: *Above the Clouds: A Berkeley View of Cloud Computing*. Technical Report, EECS Department, University of California, Berkeley, February 2009.
- [AFP*09] ARMBRUST, MICHAEL, ARMANDO FOX, DAVID PATTERSON, NICK LANHAM, HARUKI OH, BETH TRUSHKOWSKY and JESSE TRUTNA: *SCADS: Scale-independent Storage for Social Computing Applications*. In *Proceedings of the 4th Biennial Conference on Innovative Data Systems Research (CIDR 2009)*, *Online Proceedings*, Asilomar, CA, USA, January 2009. www.cidrdb.org.
- [AJNB07] ALJABER, BADER, THOMAS JACOBS, KRISHNA NADIMINTI and RAJKUMAR BUYYA: *Multimedia on Global Grids: A Case Study in Distributed Ray Tracing*. *Malaysian Journal of Computer Science*, 20(1):1–11, 2007.
- [Akam13] AKAMAI INTELLIGENT PLATFORM: *Akamai Technologies*. [Online] <http://www.akamai.com>, last accessed: June, 2013.
- [Aksa11] AKSAKALLI, I. GOEKHAN: *XMPP-based Mobile Multimedia Collaboration*. Master's thesis, RWTH Aachen University, Aachen, Germany, 2011.
- [Alap10] ALAPETITE, ALEXANDRE: *Dynamic 2D-barcodes for Multi-Device Web Session Migration Including Mobile Phones*. *Personal Ubiquitous Computing*, 14(1):45–52, 2010.
- [AlMa10] ALMEIDA, MIGUEL and ALFREDO MATOS: *Bridging the Devices with the Web Cloud: A Restful Management Architecture over XMPP*. In *Proceedings of the 6th International Mobile Multimedia Communications Conference (MOBIMEDIA 2010)*, volume 77 of *LNICST*, pages 136–150, Lisbon, Portugal, 2010. Springer.
- [AEC2] AMAZON EC2: *Amazon Elastic Compute Cloud*. [Online] <http://aws.amazon.com/de/ec2/>, last accessed: October, 2012.

BIBLIOGRAPHY

- [AmaS3] AMAZON SIMPLE STORAGE SERVICE (AMAZON S3). [Online] <http://aws.amazon.com/de/s3/>, last accessed: April, 2011.
- [AWS] AMAZON.COM, INC.: *Amazon Web Services*. [Online] <http://aws.amazon.com>, last accessed: June, 2013.
- [AHZh09] AL-MEKHLAFI, K., XIANGPEI HU and ZIGUANG ZHENG: *An Approach to Context-Aware Mobile Chinese Language Learning for Foreign Students*. In *Eighth International Conference on Mobile Business (ICMB 2009)*, pages 340–346, Dalian, Liaoning, China, June 2009. IEEE.
- [Ange06] ANGELIDES, MARIOS C.: *Multimedia Content Modeling and Personalization*. Springer, 2006.
- [Hado09] APACHE FOUNDATION: *Apache Hadoop*. [Online] <http://hadoop.apache.org>, last accessed: June 2013.
- [HDFS] APACHE FOUNDATION: *Hadoop Distributed File System*. [Online] <http://hadoop.apache.org/docs/hdfs>, last accessed: January 2013.
- [AnRa10] ANDERSON, JANNA and LEE RAINIE: *The Future of Cloud Computing*. Survey Report, Pew Research Center, 2010.
- [ATS*07] ARNDT, RICHARD, RAPHAËL TRONCY, STEFFEN STAAB, LYNDIA HARDMAN and MIROSLAV VACURA: *COMM: Designing a Well-founded Multimedia Ontology for the Web*. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC 2007 + ASWC 2007)*, pages 30–43, Busan, Korea, 2007. Springer.
- [Deli13b] AVOS SYSTEM, INC.: *Delicious - Save, Organize, and Discover Interesting Links on the Web*. [Online] <http://www.delicious.com/>, last accessed: October, 2013.
- [BaBr01] BANKO, MICHELE and ERIC BRILL: *Scaling to Very Very Large Corpora for Natural Language Disambiguation*. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL'01, pages 26–33, Toulouse, France, 2001. Association for Computational Linguistics.
- [BBC*11] BAKER, JASON, CHRIS BOND, JAMES CORBETT, J. J. FURMAN, ANDREY KHORLIN, JAMES LARSON, JEAN-MICHEL LEON, YAWEI LI, ALEXANDER LLOYD and VADIM YUSHPRAKH: *Megastore: Providing Scalable, Highly Available Storage for Interactive Services*. In *Proceedings of the 5th Biennial Conference on Innovative Data Systems Research (CIDR)*, pages 223–234, 2011.

- [BBD*08] BAILER, WERNER, LIONEL BRUNIE, MARIO DÖLLER, MICHAEL GRANITZER, RALF KLAMMA, HARALD KOSCH, MATHIAS LUX and MARC SPANIOL: *Multimedia Metadata Standards*. In FURHT, BORKO (editor): *Encyclopedia of Multimedia*, pages 568–575. Springer, 2008.
- [WebR12] BERGKVIST, ADAM, DANIEL C BURNETT, CULLEN JENNINGS and ANANT NARAYANAN: *WebRTC 1.0: Real-time Communication Between Browsers*. Working Draft, W3C, 2012.
- [BrDu00] BROWN, JOHN SEELY and PAUL DUGUID: *The Social Life of Information*. Harvard Business Press, 2000.
- [BDB*04] BALME, LIONEL, ALEXANDRE DEMEURE, NICOLAS BARRALON, JOË COUTAZ and GAËLLE CALVARY: *CAMELEON-RT: A Software Architecture Reference Model for Distributed, Migratable, and Plastic User Interfaces*. *Ambient Intelligence*, 3295:291–302, 2004. Springer.
- [Beri04] BERINGER, JOERG: *Reducing Expertise Tension*. *Communications of the ACM*, 47(9):39–40, 2004.
- [BeGr09] BENTLEY, FRANK R. and MICHAEL GROBLE: *TuVista: Meeting the Multimedia Needs of Mobile Sports Fans*. In *Proceedings of the 17th ACM International Conference on Multimedia*, MM '09, pages 471–480, Beijing, China, 2009. ACM.
- [BaHo09] BARROSO, LUIZ ANDRÉ and URS HÖLZLE: *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. *Synthesis Lectures on Computer Architecture*. Morgan & Claypool, 2009.
- [BrKo13] BRANDT, ROMAN and DEJAN KOVACHEV: *Flexible Flash Cards For Ubiquitous Micro-Learning*. In *Proceedings of 10th Conference for Informatics and Information Technology (CIIT 2013)*, pages 47–51, Pelister, Macedonia, 19–21 April 2013. University Ss Cyril and Methodius, Skopje, Macedonia. In print.
- [BMP*07] BANDELLONI, RENATA, GIULIO MORI, FABIO PATERNÒ, CARMEN SANTORO and ANTONIO SCORCIA: *Web User Interface Migration through Different Modalities with Dynamic Device Discovery*. In *Proceedings of the 2nd International Workshop on Adaptation and Evolution in Web Systems Engineering (AEWSE'07)*, pages 58–72, Como, Italy, July 2007. Springer.
- [BMZA12] BONOMI, FLAVIO, RODOLOFO MILITO, JIANG ZHU and SATEESH ADDEPALLI: *Fog Computing and Its Role in the Internet of Things*. In *Proceedings of the ACM SIGCOMM 2012 Workshop on Mobile Cloud Computing*, pages 13–16, Helsinki, Finland, 2012. ACM Press.

- [BRA11] BLUMENDORF, MARCO, DIRK ROSCHER and SAHIN ALBAYRAK: *Distributed User Interfaces for Smart Environments: Characteristics and Challenges*. In *Distributed User Interfaces CHI 2011 Workshop*, pages 25–28, Vancouver, BC, Canada, 2011. University of Castilla-La Mancha, Spain.
- [Brad11] BRADSKI, GARY: *OpenCV: Open Source Computer Vision Library*. [Online] <http://opencv.willowgarage.com/wiki/>, last accessed: September, 2011, 2011.
- [Bran12] BRANDT, ROMAN: *A Cross-platform Mobile Client for Ubiquitous mLearn*. Bachelor's Thesis, RWTH Aachen University, Aachen, Germany, 2012. Bachelor Thesis.
- [BSGi11] BOGDANOV, EVGENY, CHRISTOPHE SALZMANN and DENIS GILLET: *Contextual Spaces with Functional Skins as OpenSocial Extension*. In *ACHI 2011, The Fourth International Conference on Advances in Computer-Human Interactions*, pages 158–163, 2011.
- [BSS*05] BAILLIE, LYNNE, RAIMUND SCHATZ, RAINER SIMON, HERMANN ANEGG, FLORIAN WEGSCHEIDER, GEORG NIKLFELD and ALEXANDER GASSNER: *Designing Mona: User Interactions with Multimodal Mobile Applications*. In *Proceedings of 11th International Conference on Human-Computer Interaction (HCI International)*, pages 22–27, Las Vegas, Nevada, USA, 2005. Lawrence Erlbaum Associates.
- [BaRa08] BAEZA-YATES, RICARDO and RAGHU RAMAKRISHNAN: *Data challenges at Yahoo!*. In *15th International Conference on Extending Database Technology, EDBT '08*, pages 652–655, Nantes, France, 2008. ACM.
- [BZPr10] BURSUC, ANDREI, TITUS ZAHARIA and FRANÇOISE PRÊTEUX: *Mobile Video Browsing and Retrieval with the OVIDIUS Platform*. In *Proceedings of the International Conference on Multimedia, MM '10*, pages 1659–1662, Firenze, Italy, 2010. ACM.
- [Catt10] CATTELL, RICK: *Scalable SQL and NoSQL data stores*. *SIGMOD Rec.*, 39(4):12–27, May 2011.
- [CBC*10] CUERVO, EDUARDO, ARUNA BALASUBRAMANIAN, DAE-KI CHO, ALEC WOLMAN, STEFAN SAROIU, RANVEER CHANDRA and PARAMVIR BAHL: *MAUI: Making Smartphones Last Longer with Code Offload*. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (ACM MobiSys '10)*, pages 49–62, San Francisco, CA, USA, 2010. ACM.

- [CCJu07] CUI, YANQING, JAN. CHIPCHASE and YOUNGHEE JUNG: *Personal TV: A Qualitative Study of Mobile TV Users*. In *Proceedings of the 5th European Conference on Interactive TV: A Shared Experience*, pages 195–204, Amsterdam, The Netherlands, 2007. Springer.
- [CCS*12] CHUN, BYUNG-GON, CARLO CURINO, RUSSELL SEARS, ALEXANDER SHRAER, SAMUEL MADDEN and RAGHU RAMAKRISHNAN: *Mobius: Unified Messaging and Data Serving for Mobile Apps*. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 141–154, Low Wood Bay, Lake District, UK, 2012. ACM.
- [CDG*08] CHANG, FAY, JEFFREY DEAN, SANJAY GHEMAWAT, WILSON C HSIEH, DEBORAH A WALLACH, MIKE BURROWS, TUSHAR CHANDRA, ANDREW FIKES and ROBERT E GRUBER: *Bigtable : A Distributed Storage System for Structured Data*. *ACM Transactions on Computer Systems*, 26(2):1–26, 2008.
- [CDD*12] CHAZALET, ANTONIN, FREDERIC DANG TRAN, MARINA DESLAUGIERS, ALEXANDRE LEFEBVRE, FRANCOIS EXERTIER and JULIEN LEGRAND: *Adding Self-scaling Capability to the Cloud to meet Service Level Agreements*. *International Journal On Advances in Intelligent Systems*, 4(3 and 4):180–187, 2012.
- [CGG*13] CHANG, RUAY-SHIUNG, JERRY GAO, VOLKER GRUHN, JINGSHA HE, GEORGE ROUSSOS and WEI-TEK TSAI: *Mobile Cloud Computing Research - Issues, Challenges and Needs*. pages 442–453. IEEE Computer Society, 2013.
- [Chri09] CHRISTENSEN, JASON H.: *Using RESTful Web-services and Cloud Computing to Create Next Generation Mobile Applications*. In *Proceeding of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications (OOPSLA '09)*, pages 627–634, Orlando, FL, USA, 2009. ACM.
- [Cisc12] CISCO SYSTEMS: *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011-2016*. White paper, FLGD 10229 02/12, 2012.
- [CJK*09] CAO, YIWEI, MATTHIAS JARKE, RALF KLAMMA, OSCAR MENDOZA and SATISH SRIRAMA: *Mobile Access to MPEG-7 Based Multimedia Services*. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 102–111, Taipei, Taiwan, 2009. IEEE.

- [CJL*08] CHAIKEN, RONNIE, BOB JENKINS, PERÅKE LARSON, BILL RAMSEY, DARREN SHAKIB, SIMON WEAVER and JINGREN ZHOU: *SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets*. Proc. VLDB Endowment, 1(2):1265–1272, aug 2008.
- [CKHJ08] CAO, YIWEI, RALF KLAMMA, MIN HOU and MATTHIAS JARKE: *Follow Me, Follow You - Spatiotemporal Community Context Modeling and Adaptation for Mobile Information Systems*. In *Proceedings of the 9th International Conference on Mobile Data Management, April 27-30, 2008, Beijing, China*, pages 108–115. IEEE Society, April 2008.
- [CKJa10] CAO, YIWEI, RALF KLAMMA and MATTHIAS JARKE: *Mobile Multimedia Management for Virtual Campfire - The German Excellence Research Cluster UMIC*. International Journal on Computer Systems, Science and Engineering(IJCSSE), 25(3):251–265, May 2010.
- [CKKh09] CAO, YIWEI, RALF KLAMMA and MAZIAR KHODAEI: *A Multimedia Service with MPEG-7 Metadata and Context Semantics*. In GRIGORAS, ROMULUS, VINCENT CHARVILLAT, RALF KLAMMA and HARALD KOSCH (editors): [GCKK09], volume 441 of CEUR-WS, 2009.
- [CKKo09] CAO, YIWEI, RALF KLAMMA and DEJAN KOVACHEV: *Multimedia Processing on Multimedia Semantics and Multimedia Context*. In *Proceedings of the 10th Multimedia Metadata Community Workshop on Semantic Multimedia Database Technologies (SeMuDaTe'09)*, volume 539 of CEUR-WS, 2009.
- [CKKI10] CAO, YIWEI, DEJAN KOVACHEV, RALF KLAMMA and RYNSON W.H. LAU: *Enhancing Personal Learning Environment by Context-aware Tagging*. In *Proceedings of ICWL 2010 - Advances in Web-Based Learning, LNCS Vol. 6483*, pages 11–20, Shanghai, China, 2010. Springer.
- [CLCh07] CHEN, CHIH-MING, YI-LUN LI and MING-CHUAN CHEN: *Personalized Context-Aware Ubiquitous Learning System for Supporting Effectively English Vocabulary Learning*. In *Proceedings of the 7th IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, pages 628–630. IEEE, Jul. 2007.
- [ChMa09] CHUN, BYUNG-GON and PETROS MANIATIS: *Augmented Smartphone Applications Through Clone Cloud Execution*. In *Proceedings of the 12th Workshop on Hot Topics in Operating Systems (HotOS XII)*, Monte Verita, Switzerland, 2009. USENIX.
- [ChMa10] CHUN, BYUNG-GON and PETROS MANIATIS: *Dynamically Partitioning Applications Between Weak Devices and Clouds*. In *Proceedings of the 1st*

- ACM Workshop on Mobile Cloud Computing & Services Social Networks and Beyond (MCS '10)*, pages 1–5, San Francisco, CA, USA, 2010. ACM Press.
- [Cofe00] COFER, DAVID A.: *Informal Workplace Learning. Practice Application Brief No. 10*. [Online] <http://www.eric.ed.gov/PDFS/ED442993.pdf>, last accessed: June 2013, 2000.
- [CRB*11] CALHEIROS, RODRIGO N., RAJIV RANJAN, ANTON BELOGLAZOV, CESAR A. F. DE ROSE and RAJKUMAR BUYYA: *CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms*. *Software Practice and Experience*, 41:23–50, January 2011.
- [CRJ*10] CAO, YIWEI, D. RENZEL, M. JARKE, R. KLAMMA, M. LOTTKO, G. TOUBEKIS and M. JANSEN: *Well-Balanced Usability & Annotation Complexity in Interactive Video Semantization*. In *Proceedings of the 4th International Conference on Multimedia and Ubiquitous Engineering (MUE 2010)*, pages 1–8, August 2010.
- [ChSc08] CHEN, SHIMIN and STEVEN W. SCHLOSSER: *Map-Reduce Meets Wider Varieties of Applications*. Intel Labs Pittsburgh Tech Report, May 2008.
- [CSW*04] CHU, HAO-HUA, HENRY SONG, CANDY WONG, SHOJI KURAKAKE and MASAJI KATAGIRI: *Roam, a Seamless Application Framework*. *Journal of Systems and Software*, 69(3):209–226, 2004.
- [DBCM10] DONATO, DEBORA, FRANCESCO BONCHI, TOM CHI and YOELLE MAAREK: *Do You Want to Take Notes?: Identifying Research Missions in Yahoo! Search Pad*. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pages 321–330, Raleigh, North Carolina, USA, 2010. ACM.
- [DDC*07] DAVID, FM, BILL DONKERVOET, JC CARLYLE and EM: *Supporting Adaptive Application Mobility*. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, pages 896–905, Vilamoura, Portugal, nov 2007. Springer.
- [DCApi] DELTA CLOUD API. [Online] <http://deltacloud.apache.org/>, last accessed: October, 2013.
- [DeGh04] DEAN, JEFFREY and SANJAY GHEMAWAT: *MapReduce: Simplified Data Processing on Large Clusters*. In *Proceedings of the 6th Conference and Symposium on Operating Systems Design & Implementation (OSDI'04)*, page 10, San Francisco, CA, USA, 2004. USENIX Association.

- [DeGh08] DEAN, JEFFREY and SANJAY GHEMAWAT: *MapReduce: Simplified Data Processing on Large Clusters*. Communications of the ACM, 51(1):107–113, 2008.
- [DDJ*98] DE MICHELIS, GIORGIO, ERIC DUBOIS, MATTHIAS JARKE, FLORIAN MATTHES, JOHN MYLOPOULOS, MIKE PAPAZOUGLOU, JOACHIM W. SCHMIDT, CARSON WOO and ERIC YU: *A Three-Faceted View of Information Systems: The Challenge of Change*. Communications of the ACM, 41(12):64 – 70, December 1998.
- [DSC*08] DEMEURE, ALEXANDRE, JEAN-SÉBASTIEN SOTTET, GAËLLE CALVARY, JOËLLE COUTAZ, VINCENT GANNEAU and JEAN VANDERDONCKT: *The 4C Reference Model for Distributed User Interfaces*. In *Proceedings of the Fourth International Conference on Autonomic and Autonomous Systems*, ICAS’08, pages 61–69, Gosier, Guadeloupe, 2008. IEEE Computer Society.
- [DSL_u02] DAVIS, AGUIDO HORATIO, CHENGZHENG SUN and JUNWEI LU: *Generalizing Operational Transformation to the Standard General Markup Language*. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work (CSCW’02)*, pages 58–67. ACM Press, 2002.
- [DST*10] DANIEL, FLORIAN, STEFANO SOI, STEFANO TRANQUILLINI, FABIO CASATI, HENG CHANG and YAN LI: *MarcoFlow: Modeling, Deploying, and Running Distributed User Interface Orchestrations*. In *Proceedings of the 8th International Conference on Business Process Management Demo Track*, pages 23–27. Springer, 2010.
- [DST*12] DANIEL, FLORIAN, STEFANO SOI, STEFANO TRANQUILLINI, FABIO CASATI, CHANG HENG and LI YAN: *Distributed Orchestration of User Interfaces*. Information Systems, 37(6):539–556, 2012.
- [DCMI05] DUBLIN CORE METADATA INITIATIVE: *DCMI Metadata Terms*. <http://dublincore.org/documents/dcmi-terms/>, 2005.
- [PMet07] EBU: *P_Meta 2.0 Metadata Library*. European Broadcasting Union EBU-TECH 3295-v2, 2007.
- [ElGi89] ELLIS, CLARENCE A. and SIMON J. GIBBS: *Concurrency Control in Groupware Systems*. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pages 399–407. ACM Press, 1989.
- [EGRe91] ELLIS, CLARENCE A., SIMON J. GIBBS and GAIL REIN: *Groupware: Some Issues and Experiences*. Communications of ACM, 34:39–58, January 1991.

- [Elmq11] ELMQVIST, NIKLAS: *Distributed User Interfaces: State of the Art*. In *Distributed User Interfaces 2011*, pages 7–12, Vancouver, BC, Canada, May 2011. CHI 2011, University of Castilla-La Mancha, Spain.
- [Enco13] ENCODING.COM: *Flexible Video Transcoding Platform*. [Online] <http://www.encoding.com>, last accessed: September, 2013.
- [ESC*11] EDGE, DARREN, ELLY SEARLE, KEVIN CHIU, JING ZHAO and JAMES A. LANDAY: *MicroMandarin: Mobile Language Learning in Context*. In *Proc. of CHI'11*, pages 3169–3178, Vancouver, BC, Canada, 2011. ACM.
- [TVA05] EUROPEAN TELECOMMUNICATIONS STANDARDS INSTITUTE: *Technical Specification. Broadcast and online services: Search, select, and rightful use of content on personal storage systems (TV-Anytime)*. ETSI TS 102 822-3-1 V1.3.1, 2005.
- [JeNe10] EXPERT GROUP REPORT: *The Future of Cloud Computing. Opportunities for European Cloud Computing Beyond 2010*. Technical Report, European Commision Information Society and Media, 2010.
- [FFMp13] FFMPEG. [Online] <http://ffmpeg.org/>, last accessed: September, 2011.
- [Flic13] *The flickr Digital Photo Management System*. [Online] <http://flickr.com>, last accessed: October, 2013.
- [Flic06] *Flickr: Photo Management and Sharing Application*, 2013.
- [FLRa11] FERNANDO, NNIROSHINIE, SENG W. LOKE and WENNY RAHAYU: *Dynamic Mobile Cloud Computing: Ad Hoc and Opportunistic Job Sharing*. In *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, pages 281–286, 2011.
- [FNCa09] FOUQUET, MARC, HEIKO NIEDERMAYER and GEORG CARLE: *Cloud Computing for the Masses*. In *Proceedings of the 1st ACM Workshop on User-Provided Networking: Challenges and Opportunities*, pages 31–36, Rome, Italy, 2009.
- [FNSa01] FLINN, JASON, DUSHYANTH NARAYANAN and MAHADEV SATYA-NARAYANAN: *Self-tuned remote execution for pervasive computing*. In *Proceedings Eighth Workshop on Hot Topics in Operating Systems (HotOS)*, pages 61–66, Schloss Elamu, Germany, 2001. IEEE.
- [FoSc04] FOHRMANN, J. and E. SCHÜTTPELZ: *Die Kommunikation der Medien*. Niemeyer, Tübingen, 2004 (in German).

- [FSTS03] FLINN, JASON, SHAFEEQ SINNAMOHIDEEN, NIRAJ TOLIA and M. SATYANARYANAN: *Data Staging on Untrusted Surrogates*. In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, FAST '03, pages 15–28, San Francisco, CA, 2003. USENIX Association.
- [Sadi12] GAUTAM, SADIKSHA: *Multimedia on the Mobile Web*. Seminar Paper, RWTH Aachen University, Aachen, Germany, 2012.
- [GCB*08] GARRISS, SCOTT, RÁMON CÁCERES, STEFAN BERGER, REINER SAILER, LEENDERT VAN DOORN and XIAOLAN ZHANG: *Trustworthy and Personalized Computing on Public Kiosks*. In *Proceeding of the 6th International Conference on Mobile Systems, Applications, and Services (MobiSys '08)*, pages 199 – 210, Breckenridge, CO, USA, 2008. ACM.
- [GCKK09] GRIGORAS, ROMULUS, VINCENT CHARVILLAT, RALF KLAMMA and HARALD KOSCH (editors): *Proceedings of the 9th Workshop on Multimedia Metadata (WMM'09), Toulouse, France, March 19-20, 2009*, volume 441 of CEUR-WS, 2009.
- [Gerl07] GERLICHER, ANSGAR ROBERT SANDY: *Developing Collaborative XML Editing Systems*. PhD thesis, University of the Arts London, October 2007.
- [GGLe03] GHEMAWAT, SANJAY, HOWARD GOBIOFF and SHUN-TAK LEUNG: *The Google File System*. In *ACM SIGOPS Operating Systems Review*, volume 37, pages 29–43. ACM, 2003.
- [GJQu09] GUSTEDT, JENS, EMMANUEL JEANNOT and MARTIN QUINSON: *Experimental Validation in Large-Scale Systems: a Survey of Methodologies*. Parallel Processing Letters, page 16, 2009.
- [GKFu10] GARCIA, ADRIANA, HARI KALVA and BORKO FURHT: *A Study of Transcoding on Cloud Environments for Video Content Delivery*. In *Proceedings of the 2010 ACM Multimedia Workshop on Mobile Cloud Media Computing*, pages 13–18, Firenze, Italy, 2010. ACM.
- [GiLy02] GILBERT, SETH and NANCY LYNCH: *Brewer's Conjecture and the Feasibility of Consistent Available Partition-Tolerant Web Services*. In *ACM SIGACT News*, 2002.
- [GMG*04] GU, XIAOHUI, ALAN MESSER, IRA GREENBERG, DEJAN MILOJICIC and KLARA NAHRSTEDT: *Adaptive Offloading for Pervasive Computing*. IEEE Pervasive Computing, 3:66–73, July 2004.
- [GNC*09] GATES, A.F., O. NATKOVICH, S. CHOPRA, P. KAMATH, S.M. NARAYANAMURTHY, C. OLSTON, B. REED, S. SRINIVASAN and U. SRIVASTAVA:

- Building a High-level Dataflow System on Top of Map-Reduce: The Pig Experience*. Proceedings of the VLDB Endowment, 2(2):1414–1425, 2009.
- [GNM*03] GU, XIAOHUI, K. NAHRSTEDT, A. MESSER, I. GREENBERG and D. MILOJICIC: *Adaptive Offloading Inference for Delivering Applications in Pervasive Computing Environments*. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom 2003)*., pages 107–114, Dallas-Fort Worth, TX, USA, 2003. IEEE.
- [AIDL] GOOGLE, INC.: *Android Interface Definition Language (AIDL)*. [Online] <http://developer.android.com/guide/components/aidl.html>. last accessed: October, 2013.
- [GAE11] GOOGLE, INC.: *Google App Engine*. [Online] <http://code.google.com/appengine/>, last accessed: April 11, 2011.
- [YouT07] GOOGLE, INC.: *The YouTube Online Video Streaming Service*. [Online] <http://youtube.com>, last accessed: November, 2013.
- [GPSa10] GHIANI, GIUSEPPE, FABIO PATERNÒ and CARMEN SANTORO: *On-demand Cross-Device Interface Components Migration*. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '10)*, pages 299–308, Lisabon, Portugal, 2010. ACM Press.
- [GRJ*09] GIURGIU, IOANA, ORIANA RIVA, DEJAN JURIC, IVAN KRIVULEV and GUSTAVO ALONSO: *Calling the Cloud: Enabling Mobile Phones as Interfaces to Cloud Applications*. In *Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware (Middleware '09)*, pages 1–20, Urbana Champaign, IL, USA, nov 2009. Springer.
- [Grun94] GRUDIN, J.: *CSCW: History and Focus*. IEEE Computer, 27(5):19–26, 1994.
- [Gupt08] GUPTA, ABHISHEK KUMAR: *Challenges in Mobile Computing*. In *Proceedings of 2nd National Conference on Challenges and Opportunities in Information Technology (COIT-2008)*, pages 86–90, Mandi Gobindgarh, India, 2008. RIMT-IET.
- [GVD*11] GOVAERTS, STEN, KATRIEN VERBERT, DANIEL DAHRENDORF, CARSTEN ULLRICH, MANUEL SCHMIDT, MICHAEL WERKLE, ARUNANGSU CHATTERJEE, ALEXANDER NUSSBAUMER, DOMINIK RENZEL, MAREN SCHEFFEL, MARTIN FRIEDRICH, JOSE LUIS SANTOS, ERIK DUVAL and EFFIE L.-C LAW: *Towards responsive open learning environments: the ROLE interoperability framework*. In *Proceedings of*

- the 6th European Conference on Technology Enhanced Learning: Towards Ubiquitous Learning*, EC-TEL'11, pages 125–138. Springer-Verlag, 2011.
- [HuLe10] HUERTA-CANEPA, GONZALO and DONGMAN LEE: *A Virtual Cloud Computing Provider for Mobile Devices*. In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services Social Networks and Beyond (MCS'10)*, pages 1–5, San Francisco, CA, USA, 2010. ACM.
- [HoFe03] HÖLLERER, TOBIAS H. and STEVEN K. FEINER: *Telegeoinformatics: Location-Based Computing and Services*, chapter Mobile Augmented Reality. Taylor and Francis Books Ltd., 2004.
- [Hick11a] HICKSON, IAN: *HTML5 Web Messaging*. Working Draft, W3C, 2011.
- [Hick13] HICKSON, IAN: *The WebSocket API*. Editor's Draft, W3C, 2013.
- [HNWo08] HECKNER, MARKUS, TANJA NEUBAUER and CHRISTIAN WOLFF: *Tree, funny, to_read, google: What are Tags Supposed to Achieve? A Comparative Analysis of User Keywords for Different Digital Resource Types*. In *Proceeding of the 2008 ACM Workshop on Search in Social Media*, pages 3–10, Napa Valley, CA, USA, 2008. ACM.
- [HaSt88] HART, SANDRA G. and LOWELL E. STAVELAND: *Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research*. Human Mental Workload, 1:139–183, 1988.
- [HuSc99] HUNT, GALEN C and MICHAEL L SCOTT: *The Coign Automatic Distributed Partitioning System*. In *Proceedings of the Third Symposium on Operating System Design and Implementation (OSDI'99)*, number February, pages 187–200, New Orleans, LA, USA, 1999. USENIX Association.
- [HaTr06] HASSENZAHL, MARC and NOAM TRACTINSKY: *User Experience - A Research Agenda*. Behaviour and Information Technology, pages 91–97, 2006.
- [HTML5] *HTML5 - A vocabulary and associated APIs for HTML and XHTML*, 2011.
- [Hug05] HUG, THEO: *Micro Learning and Narration - Exploring Possibilities of Utilization of Narrations and Storytelling for the Designing of "micro units" and Didactical Micro-learning Arrangements*. In *The Fourth Media in Transition Conference (MiT4)*, Cambridge, MA, USA, 6–8 May 2005.
- [HoWa10] HORNSBY, ADRIAN and ROD WALSH: *From Instant Messaging to Cloud Computing, an XMPP review*. In *Proceedings of the The 14th IEEE International Symposium on Consumer Electronics (ISCE2010)*, Braunschweig, Germany, 2010. IEEE.

- [IBM03] IBM: *An Architectural Blueprint for Autonomic Computing*. Technical Report, 2003.
- [Real11b] IGNITE REALTIME: *Openfire XMPP Server*. [Online] <http://www.igniterealtime.org/projects/openfire/>, last accessed: April 2011.
- [Real11a] IGNITE REALTIME: *Smack XMPP Library*. [Online] <http://www.igniterealtime.org/projects/smack/>, last accessed: April 2011.
- [IgNo03] IGNAT, CLAUDIA-LAVINIA and MOIRA C. NORRIE: *Customizable Collaborative Editor Relying on TreeOPT Algorithm*. In *Proceedings of the Eighth Conference on Computer Supported Cooperative Work*, pages 315–334, Helsinki, Finland, 2003. Kluwer Academic Publishers.
- [ISO03] ISO: *Information Technology – Multimedia Content Description Interface – Part 5: Multimedia description schemes*. Technical Report ISO/IEC 15938-5:2003, International Organisation for Standardisation / International Electrotechnical Commission, May 2003.
- [ISO04d] ISO: *MPEG-7 Overview*. Technical Report, International Organisation for Standardisation, 2004.
- [ISOF10] ISO FDIS 9241-210:2010. *Ergonomics of human system interaction - Part 210: Human-centered design for interactive systems (formerly known as 13407)*. International Organization for Standardization (ISO)., 2010.
- [JHE199] JING, JIN, ABDELSALAM SUMI HELAL and AHMED ELMAGARMID: *Client-server Computing in Mobile Environments*. ACM Computing Surveys (CSUR), 31(2):117–157, 1999.
- [KAH*11] KOSTA, SOKOL, ANDRIUS AUCINAS, PAN HUI, RICHARD MORTIERAND and XINWEN ZHANG: *Unleashing the Power of Mobile Cloud Computing using ThinkAir*. CoRR, abs/1105.3232, 2011. informal publication.
- [KAK112] KOVACHEV, DEJAN, GOEKHAN AKSAKALLI and RALF KLAMMA: *A Real-time Collaboration-enabled Mobile Augmented Reality System with Semantic Multimedia*. In *Proceedings of the 2012 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pages 345–354, Pittsburgh, PA, USA, 2012. IEEE.
- [KoBe12] KOEHLER, MARTIN and SIEGFRIED BENKNER: *Design of an Adaptive Framework for Utility-Based Optimization of Scientific Applications in the Cloud*. In *Proceedings of the 2012 IEEE/ACM Fifth International Conference*

- on Utility and Cloud Computing*, UCC '12, pages 303–308. IEEE Computer Society, 2012.
- [KBD*05] KOSCH, H., L. BOSZORMENYI, M. DOLLER, M. LIBSIE, P. SCHOJER and A. KOFLER: *The Life Cycle of Multimedia Metadata*. MultiMedia, IEEE, 12(1):80–86, jan.-march 2005.
- [KCKI10] KOVACHEV, DEJAN, YIWEI CAO and RALF KLAMMA: *Augmenting Pervasive Environments with an XMPP-based Mobile Cloud Middleware*. In *Proceedings of the International Workshop on Mobile Computing and Clouds (MobiCloud 2010) in conjunction with MobiCASE 2010*, pages 361–372, Santa Clara, CA, USA, 25–28 October 2010. Springer.
- [KCKI11a] KOVACHEV, DEJAN, YIWEI CAO and RALF KLAMMA: *Mobile Cloud Computing: A Comparison of Application Models*. CoRR, abs/1107.4940, 2011.
- [KCKI11b] KOVACHEV, DEJAN, YIWEI CAO and RALF KLAMMA: *Mobile Multimedia Cloud Computing and the Web*. In *Proceedings of the IEEE Workshop on Multimedia on the Web (MMWeb 2011) in conjunction with i-Know and i-Semantics*, pages 21–26, Graz, Austria, 2011. IEEE.
- [KCKI12] KOVACHEV, DEJAN, YIWEI CAO and RALF KLAMMA: *Building Mobile Multimedia Services: A Hybrid Cloud Computing Approach*. Multimedia Tools and Applications, pages 1–30, 2012. Online first.
- [KCKI13] KOVACHEV, DEJAN, YIWEI CAO and RALF KLAMMA: *Cloud Services for Improved User Experience in Sharing Mobile Videos*. In *Proceedings of the 2013 IEEE International Symposium on Mobile Cloud, Computing and Service Engineering (MobileCloud 2013)*, pages 298–303, San Francisco, CA, USA, 2013. IEEE.
- [KCKJ11] KOVACHEV, DEJAN, YIWEI CAO, RALF KLAMMA and MATTHIAS JARKE: *Learn-as-you-go: New Ways of Cloud-Based Micro-learning for the Mobile Web*. In *Proceedings of the 10th International Conference on Web-based Learning (ICWL 2011)*, volume 7048, pages 51–61, Hong Kong, 2011. Springer.
- [KERE09] KAHEEL, AYMAN, MOTAZ EL-SABAN, MAHMOUD REFAAT and MOSTAFA EZZ: *Mobicast: A System for Collaborative Event Casting Using Mobile Phones*. In *Proceedings of the 8th International Conference on Mobile and Ubiquitous Multimedia*, pages 71–78, Cambridge, United Kingdom, 2009.

- [KGNi04] KOPONEN, TEEMU, ANDREI GURTOV and PEKKA NIKANDER: *Application Mobility with Host Identity Protocol*. In *Identifier/Locator Split and DHTs: Proceedings of the Research Seminar on Telecommunications Software*, pages 50–59, Helsinki, Finland, 2004. Helsinki University of Technology.
- [KIJa08b] KLAMMA, RALF and MATTHIAS JARKE: *Mobile Social Software for Professional Communities*. UPGRADE, IX(3):37–43, June 2008.
- [KoKI09] KOVACHEV, DEJAN and RALF KLAMMA: *Context-aware Mobile Multimedia Services in the Cloud*. In *Proceedings of the 10th International Workshop of the Multimedia Metadata Community on Semantic Multimedia Database Technologies*. Springer, 2009.
- [KoKI10] KOVACHEV, DEJAN and RALF KLAMMA: *A Cloud Multimedia Platform*. In *Proceedings of the 11th International Workshop of the Multimedia Metadata Community on Interoperable Social Multimedia Applications (WISMA-2010)*, pages 61–64, Barcelona, Spain, 19–20 May 2010. CEUR-WS.
- [KoKI12b] KOVACHEV, DEJAN and RALF KLAMMA: *Beyond the Client-Server Architectures: A Survey of Mobile Cloud Techniques*. In *Proceedings of the 2012 IEEE Workshop on Mobile Cloud Computing (MobiCC’12) held in conjunction with the 1st IEEE International Conference on Communications in China (ICCC’12)*, pages 20–25, Beijing, China, 2012. IEEE.
- [KoKI12a] KOVACHEV, DEJAN and RALF KLAMMA: *Framework for Computation Offloading in Mobile Cloud Computing*. International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI), 1(7):6–15, 2012.
- [KuLu10] KUMAR, KARTHIK and YUNG-HSIANG LU: *Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?* Computer, 43(4):51–56, April 2010.
- [Klam11] KLAMMA, RALF: *Social Software and Community Information Systems*. Habilitation, RWTH Aachen University, Aachen, Germany, 2011.
- [KNKI13] KOVACHEV, DEJAN, PETRU NICOLAESCU and RALF KLAMMA: *Mobile Real-Time Collaboration for Semantic Multimedia: A Case Study with Mobile Augmented Reality Systems*. Journal of Mobile Networks and Applications (MONET), pages 1–12, 2013. Online first.
- [Kore08] KOREN, ISTVÁN: *Conceptual Design of a mobile collaborative Platform based on Android and XMPP*. Diploma Thesis, Technische Universität Dresden, Dresden, Germany, 2008.
- [Kosc02] KOSCH, HARALD: *MPEG-7 and Multimedia Database Systems*. SIGMOD Record, 31(2):34 – 39, 2002.

- [Kosc03] KOSCH, HARALD: *Distributed Multimedia Database Technologies Supported by MPEG-7 and MPEG-21*. CRC Press, Boca Raton et al., 2003.
- [KPK*09] KEMP, ROELOF, NICHOLAS PALMER, THILO KIELMANN, FRANK SEINSTRÄ, NIELS DROST, JASON MAASSEN and HENRI BAL: *eyeDentify: Multimedia Cyber Foraging from a Smartphone*. In *Proceedings of the 11th IEEE International Symposium on Multimedia (ISM 2009)*, pages 392–399, 2009.
- [KPKB10] KEMP, ROELOF, NICHOLAS PALMER, THILO KIELMANN and HENRI BAL: *Cuckoo: a Computation Offloading Framework for Smartphones*. In *Proceedings of the 2nd International ICST Conference on Mobile Computing, Applications, and Services (MobiCASE 2010)*, Santa Clara, CA, USA, October 2010.
- [KPSV07] KNOCHE, HENDRIK, MARCO PAPALEO, M. ANGELA SASSE and ALESSANDRO VANELLI-CORALLI: *The Kindest Cut: Enhancing the User Experience of Mobile TV Through Adequate Zooming*. In *Proceedings of the 15th International Conference on Multimedia*, MULTIMEDIA '07, pages 87–96, Augsburg, Germany, 2007. ACM.
- [Krau09] KRAUTHEIM, F. JOHN: *Private Virtual Infrastructure for Cloud Computing*. In *Proceedings of the 2009 Conference on Hot Topics in Cloud Computing*, HotCloud'09, San Diego, California, 2009. USENIX Association.
- [Kris10] KRISTENSEN, MADSDARØ: *Empowering Mobile Devices Through Cyber Foraging: The Development of Scavenger, an Open, Mobile Cyber Foraging System*. PhD Thesis, Aarhus University, Aarhus, Denmark, 2010.
- [KRKC10] KOVACHEV, DEJAN, DOMINIK RENZEL, RALF KLAMMA and YIWEI CAO: *Mobile Community Cloud Computing: Emerges and Evolves*. In *Proceedings of the First International Workshop on Mobile Cloud Computing (MCC 2010)*, pages 393 – 395, Kansas City, MO, USA, 23–26 May 2010. IEEE.
- [KRNK13] KOVACHEV, DEJAN, DOMINIK RENZEL, PETRU NICOLAESCU and RALF KLAMMA: *DireWolf - Distributing and Migrating User Interfaces for Widget-Based Web Applications*. In *Proceedings of 13th International Conference on Web Engineering (ICWE 2013)*, pages 99–113, Aalborg, Denmark, 2013. Springer Berlin Heidelberg.
- [KSJ*05a] KLAMMA, RALF, MARC SPANIOL, MATTHIAS JARKE, YIWEI CAO, MICHAEL JANSEN and GEORGIOS TOUBEKIS: *ACIS: Intergenerational Community Learning Supported by a Hypermedia Sites and Monuments Database*. In GOODYEAR, P., D. G. SAMPSON, D. J.-T. YANG, KINSHUK, T. OKAMOTO, R. HARTLEY and N.-S. CHEN (editors): *Proceedings of the*

- 5th International Conference on Advanced Learning Technologies (ICALT 2005)*, July 5-8, Kaohsiung, Taiwan, pages 108–112, Los Alamitos, CA, 2005. IEEE Computer Society.
- [KSRe07] KLAMMA, RALF, MARC SPANIOL and DOMINIK RENZEL: *Community-Aware Semantic Multimedia Tagging - From Folksonomies to Commsonomies*. In TOCHTERMANN, K., H. MAURER, F. KAPPE and A. SCHARL (editors): *Proceedings of I-Media'07, International Conference on New Media Technology and Semantic Systems*, J.UCS (Journal of Universal Computer Science) Proceedings, pages 163–171, Graz, Austria, September 5–7 2007. Springer-Verlag.
- [KYK12] KOVACHEV, DEJAN, TIAN YU and RALF KLAMMA: *Adaptive Computation Offloading from Mobile Devices into the Cloud*. In *2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, pages 784–791, Madrid, Spain, 2012. IEEE.
- [Lage11] LAGESSE, BRENT J.: *Challenges in Securing the Interface Between the Cloud and Mobile Systems*. In *Proceedings of the 1th IEEE PerCom Workshop on Pervasive Communities and Service Clouds (PerCoSC 2011)*, Seattle, WA, USA, March 2011. IEEE.
- [LuCo05] LUYTEN, KRIS and KARIN CONINX: *Distributed User Interface Elements to support Smart Interaction Spaces*. In *Proceedings of the Seventh IEEE International Symposium on Multimedia, ISM '05*, pages 277–286, Irvine, California, USA, 2005. IEEE Computer Society.
- [Lech10] LECHNER, MARTIN: *ARML - Augmented Reality Markup Language*. Technical Report, Mobilizy GmbH, October 2010.
- [LGL*11] LÒPEZ-ESPIN, J. J., J. A. GALLUD, E. LAZCORRETA, A. PEÑALVER and F. BOTELLA: *A Formal View of Distributed User Interfaces*. In *Distributed User Interfaces CHI 2011 Workshop*, pages 97–100, Vancouver, BC, Canada, 2011. University of Castilla-La Mancha, Spain.
- [LiGl06] LIU, FENG and MICHAEL GLEICHER: *Video Retargeting: Automating Pan and Scan*. In *Proceedings of the 14th Annual ACM International Conference on Multimedia, MULTIMEDIA '06*, pages 241–250, Santa Barbara, CA, USA, 2006. ACM.
- [LGZh03] LIU, BING, ROBERT GROSSMAN and YANHONG ZHAI: *Mining Data Records in Web Pages*. In *Proceedings of the 9th ACM SIGKDD*, pages 601–606, Washington, D.C., 2003. ACM.
- [Li13] LI, KE: *Framework for Distributed UI of Rich Mobile/Web Applications*. Master's Thesis, RWTH Aachen University, Aachen, Germany, 2013.

- [LiMa02] LIENHART, RAINER and JOCHEN MAYDT: *An Extended Set of Haar-like Features for rapid Object Detection*. In *Proceedings of the 2002 International Conference on Image Processing*, volume 1, pages 900–903, 2002.
- [LuMa09] LUGIEZ, DENIS and STÉPHANE MARTIN: *Peer to Peer Optimistic Collaborative Editing on XML-like Trees*. [Online] <http://arxiv.org/abs/0901.4201>, last accessed: March, 2013, 2009.
- [Lome11] LOMET, DAVID: *Guest Editor’s Introduction: Cloud Data Management*. *IEEE Trans. on Knowl. and Data Eng.*, 23:1281–1281, September 2011.
- [Lott12] LOTTKO, MICHAEL: *Improving User Experience of Mobile Video Media Using Cloud Services*. Master’s Thesis, RWTH Aachen University, Aachen, Germany, 2012.
- [LRH*09] LAW, EFFIE LAI-CHONG, VIRPI ROTO, MARC HASSENZAHN, ARNOLD P.O.S. VERMEEREN and JOKE KORT: *Understanding, Scoping and Defining User Experience: A Survey Approach*. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 719–728, Boston, MA, USA, 2009.
- [LROf08] LASTOVETSKY, ALEXEY, VLADIMIR RYCHKOV and MAUREEN O’FLYNN: *A Software Tool for Accurate Estimation of Parameters of Heterogeneous Communication Models*. In *Proceedings of the 15th European PVM/MPI Users’ Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 43–54, Dublin, Ireland, 2008. Springer-Verlag.
- [LTLi10] LAGERSPETZ, EEMIL, SASU TARKOMA and TANCRED LINDHOLM: *Dessy: Search and Synchronization on the Move*. In *Proceedings of the Eleventh International Conference on Mobile Data Management (MDM 2010)*, pages 215–217, may 2010.
- [Mari09] MARINELLI, EUGENE E.: *Hyrax: Cloud Computing on Mobile Devices using MapReduce*. Master’s Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2009.
- [Math04] MATHES, ADAM: *Folksonomies — Cooperative Classification and Communication Through Shared Metadata*. <http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html>, December 2004. last accessed: October 27, 2008.
- [McBe10] MCELVANEY, JESSICA and ZANE BERGE: *Weaving a Personal Web: Using online technologies to create customized, connected, and dynamic learning*

- environments*. Canadian Journal of Learning and Technology / La revue canadienne de la apprentissage et de la technologie, 35(2), 2010.
- [MDP*00] MILOJIČIĆ, DEJAN S., FRED DOUGLIS, YVES PAINDAVEINE, RICHARD WHEELER and SONGNIAN ZHOU: *Process Migration*. ACM Computing Surveys (CSUR), 32(3):241–299, September 2000.
- [Nist09] MELL, PETER and TIM GRANCE: *The NIST Definition of Cloud Computing*, 2009.
- [MGVV09] MELCHIOR, JÉRÉMIE, DONATIEN GROLAUX, JEAN VANDERDONCKT and PETER VAN ROY: *A Toolkit for Peer-to-peer Distributed User Interfaces: Concepts, Implementation, and Applications*. In *Proceedings of the 1st ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, pages 69–78, Pittsburgh, PA, USA, 2009. ACM Press.
- [DCOM] MICROSOFT CORP.: *Distributed Component Object Model (DCOM)*. [Online] <http://msdn.microsoft.com/library/cc201989.aspx>, last accessed: August 2013.
- [MLP*06] MERA, EDISON, PEDRO LÓPEZ-GARCÍA, GERMÁN PUEBLA, MANUEL CARRO and MANUEL HERMENEGILDO: *Towards Execution Time Estimation for Logic Programs via Static Analysis and Profiling*. In *16th Workshop on Logic Programming Environments*, page 16, 2006.
- [MMES04] MULDOWNY, THOMAS, MATTHEW MILLER, RYAN EATMON and PETER SAINT-ANDRE: *XEP-0096: SI File Transfer*. XEP-0096 (Standards Track), April 2004.
- [MaPa11] MANCA, MARCO and FABIO PATERNÒ: *Distributing User Interfaces with MARIA*. In *Distributed User Interfaces 2011*, pages 93–96, Vancouver, BC, Canada, May 2011. CHI 2011, University of Castilla-La Mancha, Spain.
- [PaPh10] MAZZOLA PALUSKA, JUSTIN and HUBERT PHAM: *Interactive Streaming of Structured Data*. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 11–19, April 2010.
- [MSMe10] MILLARD, PETER, PETER SAINT-ANDRE and RALPH MEIJER: *XEP-0060: Publish-Subscribe Version 1.13, Draft*. Technical Report, XMPP Standards Foundation, 2010.
- [MVMc10] MILLER, FREDERIC P., AGNES F. VANDOME and JOHN MCBREWSTER: *Amazon Web Services*. Alpha Press, 2010.
- [Myer01] MYERS, BRAD A.: *Using Handhelds and PCs Together*. Communications of the ACM, 44(11):34–41, 2001.

BIBLIOGRAPHY

- [NCD*95] NICHOLS, DAVID A., PAVEL CURTIS, MICHAEL DIXON and JOHN LAMPING: *High-latency, Low-bandwidth Windowing in the Jupiter Collaboration System*. In *Proceedings of the 8th annual ACM symposium on User Interface and Software Technology*, UIST '95, pages 111–120, Pittsburgh, Pennsylvania, United States, 1995. ACM.
- [Niel09] NIELSEN, JAKOB: *Mobile Usability*. [Online] <http://www.useit.com/alertbox/mobile-usability.html>, last accessed: May, 2011, 2011.
- [NMH*02] NICHOLS, JEFFREY, BRAD MYERS, MICHAEL HIGGINS, JOSEPH HUGHES, THOMAS HARRIS, RONI ROSENFELD and MATHILDE PIGNOL: *Generating Remote Control Interfaces for Complex Appliances*. In *Proceedings of the 15th annual ACM Symposium on User Interface Software and Technology*, UIST '02, pages 161–170, Paris, France, 2002.
- [Nove11] NOVELL, INC: *Novell Vibe Cloud*. [Online] <https://vibe.novell.com/>, last accessed: April, 2011.
- [NoTa95] NONAKA, IKUJIRO and HIROTAKA TAKEUCHI: *The Knowledge-Creating Company*. Oxford University Press, Oxford, 1995.
- [NWG*09] NURMI, D., R. WOLSKI, C. GRZEGORCZYK, G. OBERTELLI, S. SOMAN, L. YOUSEFF and D. ZAGORODNOV: *The Eucalyptus Open-Source Cloud-Computing System*. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009 (CCGRID '09)*, pages 124–131, 2009.
- [OMVo07] O'HARA, KENTON, APRIL SLAYDEN MITCHELL and ALEX VORBAU: *Consuming Video on Mobile Devices*. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 857–866, San Jose, California, USA, 2007. ACM.
- [OnLive] ONLIVE INC.: *OnLive Game Service*. [Online] <http://www.onlive.com>, last accessed: November, 2013.
- [OpSo12] OPENSOCIAL AND GADGETS SPECIFICATION GROUP: *OpenSocial Specification 2.5.0*. [Online] <http://opensocial-resources.googlecode.com/svn/spec/2.5/>, last accessed: March, 2013.
- [Reil05] O'REILLY, TIM: *What Is Web 2.0 -Design Patterns and Business Models for the Next Generation of Software*. Technical Report, www.oreillynet.com, 2005. (July 3, 2006), <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>.

- [Oust97] OUSTERHOUT, JOHN K.: *Scripting: Higher Level Programming for the 21st Century*. IEEE Computer, 31:23–30, 1997.
- [OYZh07] OU, SHUMAO, KUN YANG and JIE ZHANG: *An Effective Offloading Middleware for Pervasive Services on Mobile Devices*. Pervasive Mob. Comput., 3:362–385, August 2007.
- [PaAs09] PARKVALL, STEFAN and DAVID ASTELY: *The Evolution of LTE towards IMT-Advanced*. Journal of Communications, 4(3), 2009.
- [PABE10] PEREIRA, RAFAEL, MARCELLO AZAMBUJA, KARIN BREITMAN and MARKUS ENDLER: *An Architecture for Distributed High Performance Video Processing in the Cloud*. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, pages 482–489, Miami, FL, USA, 2010.
- [PCRo10] PAPAKOS, PANAGIOTIS, LICIA CAPRA and DAVID S. ROSENBLUM: *VOLARE: Context-Aware Adaptive Cloud Service Discovery for Mobile Systems*. In *Proceedings of the 9th International Workshop on Adaptive and Reflective Middleware*, pages 32–38, Bangalore, India, 2010.
- [PDGQ05] PIKE, R., S. DORWARD, R. GRIESEMER and S. QUINLAN: *Interpreting the Data: Parallel Analysis with Sawzall*. Scientific Programming, 13(4):277, 2005.
- [Pear09] PEARSON, SIANI: *Taking Account of Privacy when Designing Cloud Computing Services*. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, pages 44–52, Washington, DC, USA, 2009. IEEE Computer Society.
- [Phan00] PHANOURIOU, CONSTANTINOS: *UIML: A Device-Independent User Interface Markup Language*. PhD thesis, Virginia Polytechnic Institute and State University, 2000.
- [PHE*11] PAJAK, DAWID, ROBERT HERZOG, ELMAR EISEMANN, KAROL MYSZKOWSKI and HANS-PETER SEIDEL: *Scalable Remote Rendering with Depth and Motion-flow Augmented Streaming*. Computer Graphics Forum, 30(2):415–424, 2011.
- [PiNi08] PIERCE, JEFFREY S. and JEFFREY NICHOLS: *An Infrastructure for Extending Applications’ User Experiences Across Multiple Personal Devices*. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST ’08)*, pages 101–110. ACM Press, 2008.
- [PPS*11] PATRIKAKIS, CHARALAMPOS Z., NIKOLAOS PAPAIOULAKIS, CHRYSANTHI STEFANOUDAKI, ATHANASIOS VOULODIMOS and EMMANUEL

- SARDIS: *Handling Multiple Channel Video Data for Personalized Multimedia Services: A Case Study on Soccer Games Viewing*. In *Proceedings of 7th IEEE International Workshop on Pervasive Learning, Life, and Leisure*, Seattle, USA, 2011. IEEE.
- [PSSc08] PATERNÒ, FABIO, CARMEN SANTORO and ANTONIO SCORCIA: *User Interface Migration Between Mobile Devices and Digital TV*. In *Proceedings of the 2nd Conference on Human-Centered Software Engineering and 7th International Workshop on Task Models and Diagrams*, pages 287–292, Pisa, Italy, 2008. Springer-Verlag Berlin.
- [RARo07] RELLERMEYER, JAN S., GUSTAVO ALONSO and TIMOTHY ROSCOE: *R-OSGi: Distributed Applications Through Software Modularization*. In *Proceedings of the ACM/IFIP/USENIX 8th International Middleware Conference Conference (Middleware 2007)*, pages 50–54, Newport Beach, CA, USA, November 2007. Springer.
- [RDAI09] RELLERMEYER, JAN S., MICHAEL DULLER and GUSTAVO ALONSO: *Engineering the Cloud from Software Modules*. In *Proceedings of the Workshop on Software Engineering Challenges in Cloud Computing (ICSE-Cloud, in conjunction with ICSE 2009)*, pages 32–37, Vancouver, Canada, 2009. IEEE.
- [RoMa10] ROSENBERG, JOTHY and ARTHUR MATEOS: *The Cloud at Your Service*. Manning Publications, 1st edition, 2010.
- [AMJ*09] ÅHLUND, ANDREAS, KARAN MITRA, DAN JOHANSSON, CHRISTER ÅHLUND and ARKADY ZASLAVSKY: *Context-aware Application Mobility Support in Pervasive Computing Environments*. In *Proceedings of the 6th International Conference on Mobile Technology, Application & Systems (Mobility '09)*, pages 21:1–21:4, Nice, France, September 2009. ACM.
- [RRAI08] RELLERMEYER, J, ORIANA RIVA and GUSTAVO ALONSO: *AlfredO: An Architecture for Flexible Interaction with Electronic Devices*. In *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware (Middleware 2008)*, volume 5346 of LNCS, pages 22–41, Leuven, Belgium, 2008. Springer.
- [SIP02] ROSENBERG, JONATHAN, HENNING SCHULZRINNE, GONZALO CAMARILLO, ALAN JOHNSTON, JON PETERSON, ROBERT SPARKS, MARK HANDELY, EVE SCHOOLER and OTHERS: *SIP: Session Initiation Protocol*. Technical Report, RFC 3261, Internet Engineering Task Force, 2002.
- [Sain07] SAINT-ANDRE, PETER: *Jingle: Jabber Does Multimedia*. IEEE MultiMedia, 14(1):90–94, 2007.

- [Sain08] SAINT-ANDRE, PETER: *XEP-0045: Multi-User Chat*. XMPP XEP-0045 (Standards Track), July 2008.
- [Sain11] SAINT-ANDRE, PETER: *RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*. Technical Report, XMPP Standards Foundation, 2011.
- [SAD*10] STONEBRAKER, MICHAEL, DANIEL ABADI, DAVID J. DEWITT, SAM MADDEN, ERIK PAULSON, ANDREW PAVLO and ALEXANDER RASIN: *MapReduce and parallel DBMSs: friends or foes?* Commun. ACM, 53(1):64–71, January 2010.
- [Sale13] SALESFORCE.COM, INC: *Salesforce CRM*. [Online] <http://www.salesforce.com/>, last accessed: April, 2013.
- [Sand11] SANDHOLM, THOMAS: *HP Labs Cloud-Computing Test Bed: VideoToon Demo*. [Online] http://www.hpl.hp.com/open_innovation/cloud_collaboration/cloud_demo.html, last accessed: May, 2011, 2011.
- [SSTr09] SAINT-ANDRE, PETER, KEVIN SMITH and REMKO TRON: *XMPP: The Definitive Guide: Building Real-Time Applications with Jabber Technologies*. O'Reilly Media, 1 edition, 2009.
- [Saty96] SATYANARAYANAN, MAHADEV: *Fundamental Challenges in Mobile Computing*. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 1–7, Philadelphia, PA, USA, 1996. ACM.
- [SBCD09] SATYANARAYANAN, MAHADEV, PARAMVIR BAHL, RAMÓN CÁCERES and NIGEL DAVIES: *The Case for VM-Based Cloudlets in Mobile Computing*. IEEE Pervasive Computing, 8(4):14–23, October 2009.
- [SCFe97] SULEIMAN, MAHER, MICHÈLE CART and JEAN FERRIÉ: *Serialization of Concurrent Operations in a Distributed Collaborative Environment*. In *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work (GROUP'97)*, pages 435–445. ACM, 1997.
- [SuEl98] SUN, CHENGZHENG and CLARENCE ELLIS: *Operational Transformation in Real-time Group Editors: Issues, Algorithms, and Achievements*. In *Proceedings of the 1998 ACM conference on Computer Supported Cooperative Work, CSCW '98*, pages 59–68, Seattle, Washington, United States, 1998. ACM.

- [SJZ*98] SUN, CHENGZHENG, XIAOHUA JIA, YANCHUN ZHANG, YUN YANG and DAVID CHEN: *Achieving Convergence, Causality Preservation, and Intention Preservation in Real-time Cooperative Editing Systems*. ACM Transactions on Computer-Human Interaction, 5:63–108, March 1998.
- [SKCa09] SPANIOL, MARC and RALF KLAMMA: *Media Centric Knowledge Sharing on the Web 2.0*. In *Knowledge Networks: The Social Software Perspective*, pages 46–60.
- [SKHH05] SATYANARAYANAN, MAHADEV, MICHAEL A KOZUCH, CASEY J HELFRICH and DAVID R O HALLARON: *Towards Seamless Mobility on Pervasive Hardware*. Pervasive and Mobile Computing, 1(2):157–189, July 2005.
- [SKJR06] SPANIOL, MARC, RALF KLAMMA, HOLGER JANSSEN and DOMINIK RENZEL: *LAS: A Lightweight Application Server for MPEG-7 Services in Community Engines*. In *Proceedings of I-KNOW '06, 6th International Conference on Knowledge Management, Graz, Austria, September 6–8, 2006*, pages 592–599, 2006.
- [SMPT07] SMPTE: *Metadata Dictionary Registry of Metadata Element Descriptions*. Society of Motion Picture and Television Engineers - RP210.10-2007, 2007.
- [ScRe11] SCHMIDT, RAINER and MATTHIAS RELLE: *An Approach for Processing Large and Non-uniform Media Objects on MapReduce-based Clusters*. In *Proceedings of the 13th International Conference on Asia-Pacific Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation, ICADL'11*, pages 172–181, Beijing, China, 2011. Springer-Verlag.
- [SuSh03] SULLIVAN, ARTHUR and STEVEN M. SHEFFRIN: *Economics: Principles in action*. Pearson Prentice Hall, 2003.
- [SuSu06] SUN, DAVID and CHENGZHENG SUN: *Operation Context and Context-based Operational Transformation*. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW'06)*, pages 279–288. ACM Press, 2006.
- [SSSc10] SCHUSTER, DANIEL, THOMAS SPRINGER and ALEXANDER SCHILL: *Service-based Development of Mobile Real-time Collaboration Applications for Social Networks*. In *Proceedings of IEEE PerCom Workshops (PerCol'10)*, pages 232–237, 2010.
- [SSWi07] SCHIERL, THOMAS, THOMAS STOCKHAMMER and THOMAS WIEGAND: *Mobile Video Transmission Using Scalable Video Coding*. Circuits and Systems for Video Technology, IEEE Transactions on, 17(9):1204–1217, sept. 2007.

- [Stan11] STANFORD MOBILE AND SOCIAL COMPUTING RESEARCH GROUP: *Junction Documentation for Application Developers*. [Online] <http://mobisocial.stanford.edu/index.php?page=junction>, last accessed: March, 2013.
- [Stoc11] STOCKHAMMER, THOMAS: *Dynamic Adaptive Streaming over HTTP: Standards and Design Principles*. In *Proceedings of the second annual ACM conference on Multimedia systems*, MMSys '11, pages 133–144, San Jose, CA, USA, 2011. ACM.
- [STVa05] SHARPLES, MIKE, JOSIE TAYLOR and GIASEMI VAVOULA: *Towards a Theory of Mobile Learning*. pages 1–9, 2005.
- [STWD10] SONG, WEI, DIAN W. TJONDRONEGORO, SHU-HSIEN WANG and MICHAEL J. DOCHERTY: *Impact of Zooming and Enhancing Region of Interests for Optimizing User Experience on Mobile Sports Video*. In *Proceedings of the International Conference on Multimedia*, pages 321–330, Firenze, Italy, 2010. ACM.
- [SVPS11] SUMMA, B., H.T. VO, V. PASCUCCI and C. SILVA: *Massive Image Editing on the Cloud*. In *Robotics and Applications with Symposia 739: Computational Photography and 740: Computer Vision*. ACTA Press, 2011.
- [SuYi07] SUBRAMANYA, S.R. and B.K. YI: *Enhancing the User Experience in Mobile Phones*. *Computer*, 40(12):114–117, December 2007.
- [TedV13] TED CONFERENCES, LLC: *TED - Ideas worth spreading*. [Online] <http://www.ted.com/>, last accessed: October, 2013.
- [HTTo09] TONY HEY, STEWART TANSLEY and KRISTIN TOLLE (editors): *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.
- [TuPe91] TURK, MATTHEW and ALEX PENTLAND: *Eigenfaces for Recognition*. *J. Cognitive Neuroscience*, MIT Press, 3:71–86, January 1991.
- [TSJ*09] THUSOO, A., J.S. SARMA, N. JAIN, Z. SHAO, P. CHAKKA, S. ANTHONY, H. LIU, P. WYCKOFF and R. MURTHY: *Hive: A Warehousing Solution Over a Map-Reduce Framework*. *Proceedings of the VLDB Endowment*, 2(2):1626–1629, 2009.
- [VCHu03] VETRO, ANTHONY, CHARILAOS CHRISTOPOULOS and HUIFANG SUN: *Video Transcoding Architectures and Techniques: An Overview*. *IEEE Signal Processing Magazine*, 20(2):18 – 29, mar 2003.
- [Vetr11] VETRO, ANTONY: *The MPEG-DASH Standard for Multimedia Streaming*. *IEEE MultiMedia*, 18(4):62–67, 2011.

- [Voig09] VOIGT, MICHAEL: *Erweiterung und Anpassung des Collaborative Editing Framework for XML (CEFX)*. Master's thesis, University of Applied Sciences Erfurt, 2009.
- [VRCL09] VAQUERO, LUIS M, LUIS RODERO-MERINO, JUAN CACERES and MAIK LINDNER: *A Break in the Clouds: Towards a Cloud Definition*. SIGCOMM Computer Communication Review, 39(1):50–55, 2009.
- [VVLC05] VANDERVELPEN, CHRIS, GEERT VANDERHULST, KRIS KRIS LUYTEN and KARIN CONINX: *Light-Weight Distributed Web Interfaces: Preparing the Web for Heterogeneous Environments*. In *Proceedings of the 5th International Conference on Web Engineering*, volume 3579 of LNCS, pages 197–202, Sydney, Australia, 2005. Springer-Verlag Berlin.
- [Wal05] VANDER WAL, THOMAS: *Designing for the Personal InfoCloud*. [Online] <http://de.slideshare.net/vanderwal/designing-for-personal-infocloud> last accessed: May 2013.
- [VoZh09] VOAS, JEFFREY and JIA ZHANG: *Cloud Computing: New Wine or Just a New Bottle?* IT Professional, 11(2):15–17, 2009.
- [OMRe11] W3C VIDEO ON THE WEB ACTIVITY: *Ontology for Media Resources 1.0*. [Online] <http://www.w3.org/TR/2011/CR-mediaont-10-20110707/>, last accessed: November, 2011, July 2011.
- [Wang11] WANG, XUAN: *Layar Augmented Reality Browser*. [Online] <http://layar.pbworks.com/>, last accessed: April 2011.
- [WEE*08] WILHELM, REINHARD, JAKOB ENGBLOM, ANDREAS ERMEDAHL, NIKLAS HOLSTI, STEPHAN THESING, DAVID WHALLEY, GUILLEM BERNAT, CHRISTIAN FERDINAND, REINHOLD HECKMANN, TULIKA MITRA, FRANK MUELLER, ISABELLE PUAUT, PETER PUSCHNER, JAN STASCHULAT and PER STENSTROEM: *The Worst-Case Execution-Time Problem, Overview of Methods and Survey of Tools*. ACM Trans. Embed. Comput. Syst., 7:36:1–36:53, May 2008.
- [Weis91] WEISER, MARK: *The Computer for the 21st Century*. Scientific American, 265(3):94–104, 1991.
- [Weng98] WENGER, ETIENNE: *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press, Cambridge, UK, 1998.
- [WSG*09] WOOD, TIMOTHY, ALEXANDRE GERBER, K. K. RAMAKRISHNAN, PRASHANT SHENOY and JACOBUS VAN DER MERWE: *The Case for*

- Enterprise-ready Virtual Private Clouds*. In *Proceedings of the 2009 conference on Hot topics in cloud computing*, HotCloud'09, San Diego, California, 2009. USENIX Association.
- [WHMa10] WU, HUAIGU, L HAMDI and N MAHE: *TANGO: A Flexible Mobility-Enabled Architecture for Online and Offline Mobile Enterprise Applications*. In *Proceedings of 11th International Conference on Mobile Data Mangement (MDM 2010)*, pages 230–238, Kanas City, MO, USA, 2010. IEEE.
- [x26406] WILSON, REED: *x264farm - A Distributed Video Encoder*. [Online] <http://omion.dyndns.org/x264farm/x264farm.html>, last accessed: July, 2013.
- [WMLa10] WANG, DAVID, ALEX MAH and SOREN LASSEN: *Google Wave Operational Transformation*. Whitepaper, Google, Inc, July 2010.
- [Wowz12] WOWZA MEDIA SYSTEMS. [Online] <http://www.wowza.com/>, last accessed: October, 2012.
- [WYLD10] WHITE, BRANDYN, TOM YEH, JIMMY LIN and LARRY DAVIS: *Web-scale Computer Vision using MapReduce for Multimedia Data Mining*. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining, MDMKDD '10*, pages 9:1–9:10, Washington, D.C., 2010. ACM.
- [XenH13] XEN.ORG: *Xen Hypervisor*. [Online] <http://www.xen.org/products/xenhyp.html>, last accessed: January, 2013.
- [XSC*04] XIA, STEVEN, DAVID SUN, CHENGZHENG SUN, DAVID CHEN and HAIFENG SHEN: *Leveraging Single-user Applications for Multi-user Collaboration: The Coword Approach*. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW'04)*, pages 162–171. ACM Press, 2004.
- [YBSe10] YOU, YU, PETROS BELIMPASAKIS and PETRI SELONEN: *A Hybrid Content Delivery Approach for a Mixed Reality Web Service Platform*. In YU, ZHIWEN, RAMIRO LISCANO, GUANLING CHEN, DAQING ZHANG and XINGSHE ZHOU (editors): *Ubiquitous Intelligence and Computing*, volume 6406 of *LNCS*, pages 563–576. Springer, 2010.
- [YKSG94] YANG, J., A. KHOKHAR, S. SHEIKH and A. GHAFOR: *Estimating Execution Time For Parallel Tasks in Heterogeneous Processing (HP) Environment*. In *Heterogeneous Computing Workshop, 1994., Proceedings*, pages 23–28, apr 1994.

- [Yu97b] YU, ERIC S. K.: *Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering*. In *Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97) Jan. 6-8, 1997, Washington D.C., USA*, pages 226 – 235, 1997.
- [Yu97a] YU, ERIC S. K.: *Why Agent-Oriented Requirements Engineering*. In DUBOIS, E., A.L. OPDAHL and K. POHL (editors): *Proceedings of 3rd International Workshop on Requirements Engineering: Foundations for Software Quality (June 16-17, 1997, Barcelona, Catalonia)*. Presses Universitaires de Namur, 1997.
- [Yu11] YU, TIAN: *Towards Augmenting Mobile Devices Through Cloud Services*. Diploma Thesis, RWTH Aachen University, Aachen, Germany, 2011.
- [ZCB010] ZHANG, QI, LU CHENG and RAOUF BOUTABA: *Cloud Computing: State-of-the-art and Research Challenges*. *Journal of Internet Services and Applications*, 1:7–18, 2010.
- [Zenc13] ZENCODER INC.: *Zencoder.com - cloud-based video and audio encoding software as a service*. [Online] <http://www.zencoder.com>, last accessed: September, 2013.
- [ZIBu11] ZUZAK, IVAN, MARKO IVANKOVIC and IVAN BUDISELIC: *A Classification Framework for Web Browser Cross-Context Communication*. CoRR, abs/1108.4770, 2011.
- [ZJKG10] ZHANG, XINWEN, SANGO JEONG, ANUGEETHA KUNJITHAPATHAM and SIMON GIBBS: *Towards an Elastic Application Model for Augmenting Computing Capabilities of Mobile Platforms*. In *The Third International ICST Conference on MOBILE Wireless MiddleWARE, Operating Systems, and Applications*, pages 270–284, Chicago, IL, USA, 2010. Springer.
- [ZKJG11] ZHANG, XINWEN, ANUGEETHA KUNJITHAPATHAM, SANGO JEONG and SIMON GIBBS: *Towards an Elastic Application Model for Augmenting the Computing Capabilities of Mobile Devices with Cloud Computing*. *Mobile Networks and Applications*, 16:270–284, 2011.
- [ZLWL11] ZHU, WENWU, CHONG LUO, JIANFENG WANG and SHIPENG LI: *Multimedia Cloud Computing*. *Signal Processing Magazine*, 28(3):59–69, May 2011.
- [ZSG*09] ZHANG, XINWEN, JOSHUA SCHIFFMAN, SIMON GIBBS, ANUGEETHA KUNJITHAPATHAM and SANGO JEONG: *Securing Elastic Applications on Mobile Devices for Cloud Computing*. In *CCSW '09: Proceedings of the 2009 ACM Workshop on Cloud Computing Security*, pages 127–134, Chicago, IL, USA, November 2009. ACM.

- [ZTQ*10] ZHANG, LIDE, BIRJODH TIWANA, ZHIYUN QIAN, ZHAOGUANG WANG, ROBERT P. DICK, ZHUOQING MORLEY MAO and LEI YANG: *Accurate On-line Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones*. In *Proceedings of the eighth IEEE/ACM/IFIP International Conference on Hardware/software Codesign and System Synthesis*, CODES/ISSS '10, pages 105–114, Scottsdale, Arizona, USA, 2010. ACM.
- [ZWWe12] ZHONG, LONGZHAO, BEIZHAN WANG and HAIFANG WEI: *Cloud Computing Applied in the Mobile Internet*. In *7th International Conference on Computer Science Education (ICCSE 2012)*, pages 218–221, Melbourne, Australia, 2012. IEEE.

List of Figures

1.1	Research methods employed in the dissertation	5
2.1	Evolution of IT leading to cloud computing	12
2.2	Capacity versus utilization with cloud and traditional infrastructure	14
2.3	“X” as a Service	18
2.4	Software architecture of a virtualized server	21
2.5	Physical organization in a datacenter	22
2.6	Hadoop distributed file system architecture	23
2.7	The flow of a MapReduce computation	24
2.8	Media cloud and cloud media	27
2.9	Basic mobile cloud architecture	31
2.10	MPEG-7 Multimedia Description Schemes	33
2.11	Information system key components for communities of practice	43
2.12	Media centric theory of learning in communities of practice	46
3.1	Related work at the intersection of the main research ares	50
3.2	The Split&Merge approach of video encoding	51
3.3	Audiovisual data processing using MapReduce	53
3.4	Overview of a MPEG-DASH streaming system	54
3.5	CloneCloud categories for augmented execution	58
3.6	Dynamic virtual machine synthesis timeline	59
3.7	AlfredO architecture	60
3.8	Reference architecture for elastic applications	61
3.9	SeViAnno - A Web application for multimedia semantic annotation	74
4.1	i* Requirements model of multimedia “cloudified” server	89

LIST OF FIGURES

4.2	MAPE-K loop for cloud self-automated management	93
4.3	A diagram of the offloading process during cloud-based augmentation . .	95
4.4	Cost model of elastic mobile cloud applications	96
4.5	Simple architecture for Fog computing	101
4.6	Feature comparison between the three mobile cloud computing models . .	102
5.1	Layered architecture of i5Cloud	110
5.2	State machine diagram of cloud computing instances	112
5.3	Mobile offloading middleware architecture	114
5.4	Intelligent and fast video adaptation cloud services	117
6.1	Cloud monitoring: videos	125
6.2	Cloud monitoring: processing status	125
6.3	Cloud monitoring: CPU load	126
6.4	Transcoding execution time: i5Cloud private cloud and Amazon EC2 . .	126
6.5	Scalability of i5Cloud	127
6.6	CloudSim simulation of Cloud Video Transcoder	128
6.7	MVCS scenario workflow	130
6.8	Segment and metadata based video stream browsing	131
6.9	MVCS use case diagram	132
6.10	User interface of a MVCS-based mobile application	133
6.11	Video processing workflow	134
6.12	NASA TLX user workload	136
6.13	NASA TLX user workload at zoomed videos	136
6.14	NASA TLX user workload at video browsing	137
6.15	Comparison of cloud processing time	138
6.16	MACS prototype application	140
6.17	CPU executed instructions of N-queens	141
6.18	Execution time of N-queens	142
6.19	Total time distribution of N-queens	143
6.20	Energy consumption of N-queens	144
6.21	CPU executed instructions of face detection	145
6.22	Execution time of face detection	146

6.23	Energy consumption of face detection	146
6.24	Total time distribution of face detection:	148
6.25	Screen snapshots of AnViAnno	150
6.26	XMPP-based mobile multimedia collaboration	151
6.27	Main components of a mobile client for collaborative multimedia processes	152
6.28	Sequence diagram of collaborative annotation	153
6.29	XmmC ER diagram	156
6.30	XmmC MAR browser	157
6.31	UI sequence of actions for collaborative annotation	158
6.32	Web application for collaborative annotation	159
6.33	Execution time of remote concurrent XML editing operations	162
6.34	Location mapping of historical artifacts at evaluation session	164
6.35	Increase of cultural heritage awareness	165
6.36	Main aspects of personal clouds for learning	167
6.37	Micro-learning information life-cycle	169
6.38	Micro-learn workflow	172
6.39	Micro-learn evaluation results	173
6.40	DireWolf distribution of widgets	176
6.41	Common widget UI (a) versus a distributed widget UI approaches (b) . . .	181
6.42	Requirements to a dynamic widget-based DUI framework	182
6.43	Abstract architecture of the DUI framework	184
6.44	Sequence diagram of the continuous migration of active widgets	185
6.45	DUI manager user interface in a widget space sidebar panel	187
6.46	Three different widgets used in the DireWolf user preference evaluation .	190
6.47	Results of the evaluation using SeViAnno 2.0	192
6.48	Mapping of CAELUS-based prototypes to the respective facets	196

List of Tables

2.1	Key features of Web 2.0 and cloud computing	44
2.2	Mapping between media-theoretic operations and cloud services	47
3.1	Comparison of existing and proposed mobile cloud computing approaches	66
4.1	Summary of the three facets and their sub-perspectives	86
4.2	Application classes and optimal cloud models	102
5.1	Comparison of CAELUS with related approaches	120
6.1	Hardware profile details of the i5Cloud and Amazon EC2 instances . . .	125
6.2	Hardware components of mobile device and cloud node	140
6.3	Estimated energy consumption of mobile device	141
6.4	Video duration and file size	144
6.5	Video duration and speedup of face detection in video file	145
6.6	Video duration and energy savings	146
6.7	Mobile devices used at XMMC evaluation session	164
6.8	XMMC user evaluation questionnaire results	165

Appendix A

List of Abbreviations

Abbreviation	Description	First mentioned
AIDL	Android Interface Definition Language	p. 113
API	Application Programmable Interface	p. 18
AR	Augmented Reality	p. 155
ARML	Augmented Reality Markup Language	p. 34
AWS	Amazon Web Services	p. 11
CAELUS	Cloud Architecture for Enabling Mobile Multimedia Services	p. iii
CAM	Collaborative Annotation Module	p. 153
CDN	Content Delivery Network	p. 27
CEFX	Collaborative Editing Framework	p. 69
CMAX	Consistency Maintenance Algorithm for XML	p. 68
CoP	Communities of Practice	p. 42
DASH	Dynamic Adaptive Streaming over HTTP	p. 39
DBMS	Database Management System	p. 25
DDL	Descriptor Definition Language	p. 25
DFS	Distributed File System	p. 23
DOM	Document Object Model	p. 162
DUI	Distributed User Interface	p. 175
EC2	Elastic Cloud Compute	p. 18
GFS	Google System	p. 23
GPS	Global Positioning System	p. 157
HD	High Definition	p. 3
HDFS	Hadoop Distributed File System	p. 23

HTML	Hypertext Markup Language	p.	57
HTTP	Hypertext Transfer Protocol	p.	37
IaaS	Infrastructure as a Service	p.	17
ILP	Integer Linear Programming	p.	99
IPC	Inter-process Communication	p.	115
ISO	Organization for Standardization	p.	39
ISR	Internet Suspend/Resume	p.	62
IWC	Inter-widget Communication	p.	180
LAS	Lightweight Application Server	p.	72
MACS	Mobile Augmentation Cloud Services	p.	113
MACS-FTM	MACS File Transfer Manager	p.	115
MAPE-K	Monitor, Analyze, Plan, Execute and Knowledge (Loop)	p.	91
MAR	Mobile Augmented Reality	p.	155
MCC	Mobile Cloud Computing	p.	28
MDS	MPEG-7 Multimedia Description Schemes	p.	33
MEX	Mobile User Experience	p.	40
MPEG-7	Multimedia Content Description Interface	p.	33
MRTC	Mobile Real-time Collaboration	p.	155
MSP	Media Service Provider	p.	27
MVCS	Mobile Video Cloud Services	p.	128
NASA-TLX	NASA (National Aeronautical Space Agency) Task Load Index	p.	135
OCR	Optical Character Recognition	p.	171
PCL	Personal Clouds for Learning	p.	168
PLE	Personal Learning Environment	p.	168
PaaS	Platform as a Service	p.	18
P2P	Peer-to-Peer	p.	27
REST	Representational State Transfer	p.	37
PKM	Personal Knowledge Management	p.	167
ROI	Region of Interest	p.	41
ROLE	Responsive Open Learning Environments	p.	186
RPC	Remote Procedure Call	p.	115
RTP	Real Time Transport Protocol	p.	38
RTSP	Real Time Streaming Protocol	p.	38

RTP	Round Trip Time	p.	91
SaaS	Software as a Service	p.	19
SECI	Socialization, Externalization, Internalization, and Combination (Knowledge creation operations)	p.	45
SIP	Session Initiation Protocol	p.	86
SOA	Service Oriented Architecture	p.	72
SOAP	Simple Object Access Protocol	p.	37
SVG	Scalable Vector Graphics	p.	119
QoE	Quality of Experience	p.	39
QoS	Quality of Service	p.	26
OT	Operational Transformation	p.	68
VC	Virtual Campfire	p.	73
VM	Virtual Machine	p.	17
UX	User Experience	p.	39
W3C	World Wide Web Consortium	p.	202
XEP	XMPP Extension Protocol	p.	36
XML	Extensible Markup Language	p.	33
XMPP	Extensible Messaging and Presence Protocol	p.	36

Appendix B

Curriculum Vitae

Name: Dejan Kovachev

Birthday: January 26, 1985

Birth Place: Strumica, Macedonia

Address: Roermonder Str. 34
52072 Aachen

Phone: 01578 7817596

Email: kovachev@dbis.rwth-aachen.de



School: September 1991 - June 1999: Primary School “Mosha Piade”,
Strumica, Macedonia
September 1999 - June 2003: High School “Jane Sandanski”,
Strumica, Macedonia

Language Skills: Macedonian (native)
English (fluent)
German (very good)

Academic Education: September 2003 - June 2008: “Sts Cyril and Methodius” University,
Skopje, Macedonia
Major in Computer Science and Computer Engineering
Minor in Automation Engineering
June 2008: Diploma in Electrical Engineering and Information
Technology (Dipl.-Ing.)

Professional Experiences: Research scholar supported by B-IT (Bonn-Aachen International
Graduate Center for Information Technology) at the Chair for Infor-
mation Systems and Databases (Prof. Dr. M. Jarke) for the period
from May 2009 till April 2012; research assistant in the German
Excellence Research Cluster “UMIC” since May 2012.