MaaS – Microscopy as a Service

Kuckelberg, Alexander[1,a], Janecek, David[1,b]

[1]VIA Consulting & Development GmbH, Römerstr. 50, 52064 Aachen
[a]a.kuckelberg@via-con.de, ++49 241 463662-16
[b]d.janecek@via-con.de, ++49 241 463662-19

1 Introduction

Microscopic railway operations research tools and approaches have been studied, realized and implemented throughout the last decades. They are in daily use at several infrastructure managers, railway undertakings, authorities, consultants, planning and construction engineers, transport associations etc.

They offer microscopic running time calculation, detailed occupation computation, conflict detection, conflict prognosis, dispatching support, railway simulation, capacity assessment, vehicle and staff planning, timetable scheduling and much more, often theoretically overbred and highly developed up to seconds.

While multiple microscopic railway operations research tools are dedicated to specific tasks, operational environments or actions, their core functionality can partially be identified and encapsulated within modules and functional units. As a worthwhile next step, they can be published and provided as autonomous services in service oriented architectures to support several, heterogeneous clients with comparable and reliable computational calculations.

This paper identifies functionalities worth to be offered as a service and lines out approaches to separate and encapsulate these microscopic functionalities, to define interfaces to call and use them and to provide them as services to be used in a distributed, transparent manner. The resulting services presented are *LaaS* – LUKS-as-a-Service and its successor *MaaS* – Microscopy-as-a-Service.

In section 1, the paper motivates microscopic services, Section 2 focusses on necessary input data and possible underlying data models, illustrated by figures of the microscopic railway operations research tool LUKS® that implements many corresponding data models and is therefore used as basis for the *LaaS* realization. Section 3 introduces service interfaces of the microscopic service; Section 4 presents the actual Proof-of-Concept

(PoC) implementation of *LaaS*. The practical experiences as well as further extensions of the PoC result in the extended implementation *Microscopy as a Service* - *MaaS* - presented in section 5. Section 6 finally concludes.

2 Macroscopic vs. Microscopic Approaches

Microscopic railway operations research tools are used for a wide range of different tasks and offer many different functionalities. They are built upon very detailed data structures compared to macroscopic tools, which allows in-depth calculations of many details.

Macroscopic tools on the other side use much simpler data structures, but distinguish at least stations and lines in-between. Depending on the granularity of the respective macroscopic tool, it may handle further information about the length of single lines, the number of tracks, permitted speeds or other infrastructure details. Parameters of stations might be the number of available tracks, track lengths, reachability or route exclusions.

Besides the infrastructure, the modelling of train information also differs between the approaches. Microscopic tools may use sophisticated data including time, position and time supplements for each element along the way of a single train. Using the underlying infrastructure model, they are able to calculate running times, the braking behaviour based on different protection systems or blocking times. Macroscopic tools on the other side provide a simple running time calculation resp. estimation or accesses pre-calculated running times.

While both approaches are fitting different dedicated requirements and have strength with respect to specific usage scenarios, a model overcome might be promising to couple strengths from both worlds. The approach *LaaS* (LUKS-as-a-Service) introduced in this paper extracts functionalities from the microscopic tool LUKS® and provides them in a service orientated manner. The functionalities were used successfully by a macroscopic tool which succeeded this proof of concept.

The services are suitable for different granularity and detail levels, e.g. detail of route definition or completeness of train characteristics. This flexibility supports a wide usage range of the prototypical *LaaS* implementation. Focused on a simple interface usage, the following microscopic functionalities were identified worth to be extracted and provided in the proof of concept, demonstrating the potential of this service oriented approach:

- Routing services: Given an arbitrary sequence of initial station names, the routing service is able to derive a valid sequence of neighboring and (microscopically) consistently connected stations. The routing considers route priorities, usage characteristics and route lengths when selecting the best fitting train route.

- Master data provision: For succeeding calculation services, *LaaS* holds tractive units and parameters, which allows requesting clients to refer to tractive units by only their name; further knowledge about tractive units is not necessary.
- Basic running time calculation: Based on a train course and basic information about the train and its tractive unit, a simple running time calculation can be executed using default values e.g. for braking deceleration or running time supplements. A set of computations can be executed within one calculation request.
- Extended running time calculation: Depending on the grade of details of the underlying infrastructure, the abilities of the selected train set and the provided parameters, the running time calculation is able to consider many additional influences like specific train protection and control system (e.g. when stopping for a closed signal or braking towards speed restrictions), consider the maximum allowed speed based on corresponding braking tables and much more.

Based on running time calculations and blocking time theory, microscopic occupation computations and conflict detection can be requested:

- Occupation computation: For a valid running time calculation occupation block computation along the train course reflecting the usage of infrastructure in time can be requested. Detailed information about each block is provided to the service caller supporting external and capsulated blocking time theory usage.
- Buffer time determination: For several blocking time stairways of different train runs additionally buffer times between the occupations can be requested as well as the location of the lowest times. This information can e.g. support macroscopic timetabling processes.
- Conflict detection: Negative buffer times between train runs are denoted as conflicts and constitute a concurrent usage of infrastructure by different train sets. Based on the detailed infrastructure data, *LaaS* is able to identify these conflicts. In this way, macroscopic caller are assisted in detailed timetabling without assumptions, especially in stations and station heads.

These services supports macroscopic applications as demonstrated by *LaaS*, but also enables the realization of advanced railway operations research approaches like simulations, analytics or capacity assessment accordingly e.g. UIC 406 without the necessity to implement these functionalities separately.

3 Microscopic Data Model

The provision of the desired microscopic functionalities through a service requires a technical basis built upon suitable data structures and models. These structures and models are required to support microscopic granularities like infrastructure mileages at a meter based accuracy and running time determination at second accuracy.

A sufficient microscopic data model suitable for the intended functionalities comprises of at least the following domains: infrastructure, master data and train data.

3.1 Infrastructure

Infrastructure data is the basis of all microscopic functionalities like routing or running time calculations. Therefore, the data quality is crucial for all calculation qualities. All infrastructure elements must provide valid mileages to allow distance calculations.

An elementary infrastructure only contains topology information like switches, crossings, crossing switches, track ends and tracks in-between supporting simple running time calculations (Figure 1a).

Several elements enhance the infrastructure and add further information increasing accuracy of running time calculations seriously (extract): speed restrictions, gradients, stopping positions, areas for train protection systems, running time measuring points, tunnels, curves etc. (Figure 1b).

Additionally, the microscopic occupation computation is enabled and improved by the following elements: Main- and pre-signals, marker boards, liberation equipment like axle counters as well as special elements for extended occupation evaluation like encounter restrictions (tunnel) or platform accesses (Figure1c).

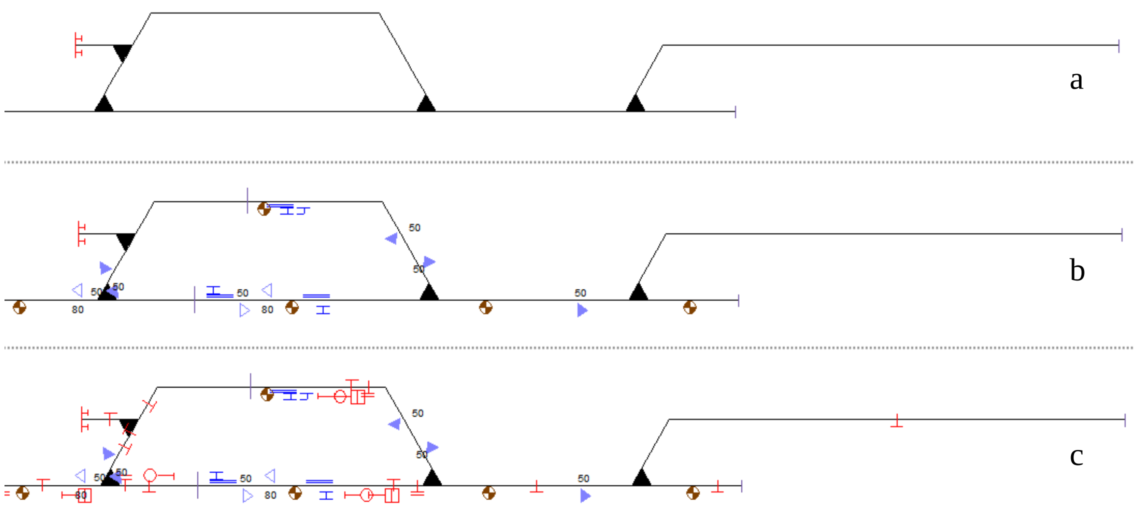

Figure 13: Microscopic infrastructure granularities

Another covering layer with routes (itineraries) additionally defines interlocking related constraints defining the usability of infrastructures. In any way, finally train runs are represented and modelled by a coherent list of infrastructure elements.

3.2 Master Data

Microscopic master data may consist of several sub-domains from which tractive units and braking tables should be mentioned explicitly in the following.

Tractive unit data embraces obvious data like a (standardised) name, length, weight or maximum speeds as well as dynamic (physical) information for acceleration (traction power data, Figure 14) and deceleration processes (braking accelerations).

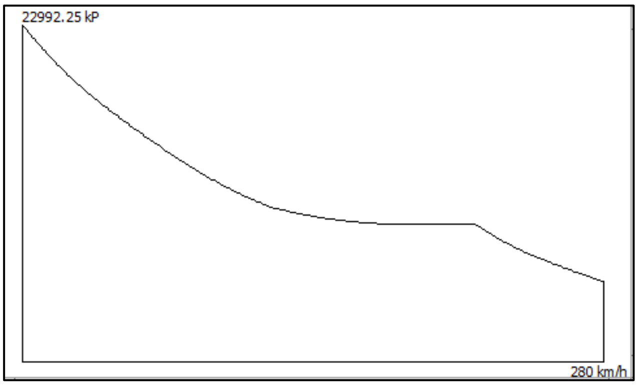

Figure 14: Microscopic traction power of a single tractive unit

Braking tables on the other hand are used to restrict the permitted maximum speed if braking percentages are missing. In combination with the braking abilities of single train sets, a microscopic tool can identify infrastructure areas, in which the speed has to be lowered to ensure a safe operation.

3.3 Train Data

The definition of train runs utilises the previously introduced data structures: active tractive units (from master data), information concerning train length, weight, diverging maximum speeds, train protection systems available, train kind etc. Train courses finally define a coherent list of infrastructure elements (on behalf of stations, routes etc.); they are additionally amended with arrival and departure times, minimum and scheduled dwell times and time margin distributions to calculate a precise running time for a train run as shown in Figure 15 (including different speed profiles, acceleration and deceleration, ETCS guidance curves etc.).

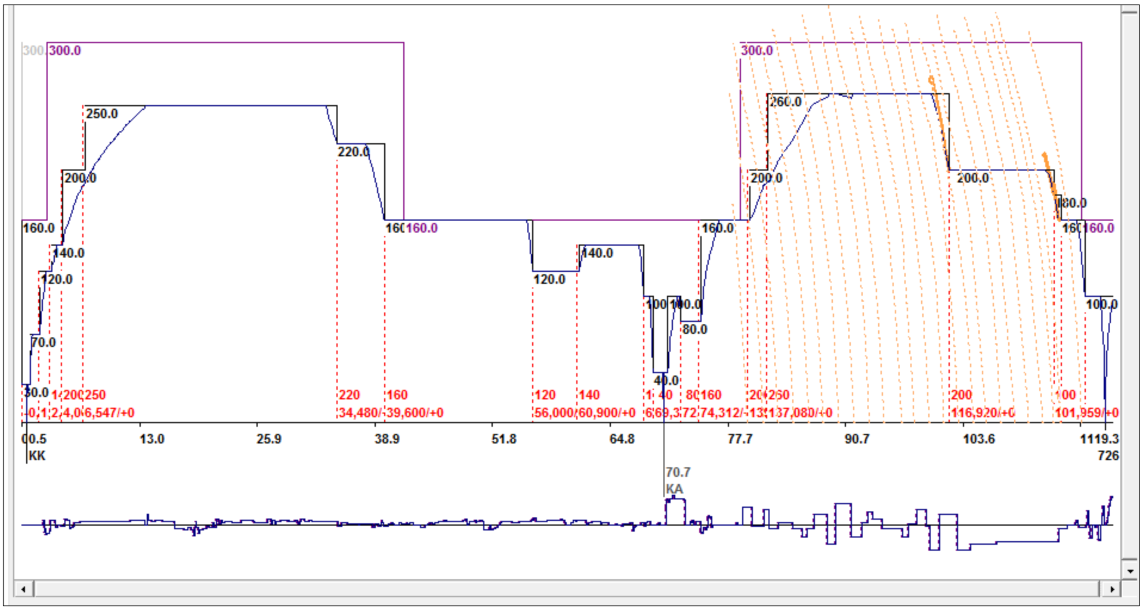

Figure 15: Velocity diagram of train run along its path

The *LaaS* approach presented in the following directly relies on these microscopic information, which are derived and accessed on behalf of current LUSK® databases. Therefor using LUKS® as microscopic basis, all infrastructure import interfaces and modification functionalities can be performed “in a process nutshell”.

Additionally, using a consistent infrastructure database as the one provided by LUKS® enables *LaaS* to hold infrastructure information in memory and to reduce requests to route indicators (instead of complete infrastructure lists when requesting), master data and traction unit references which seriously reduces submitted data quantities and speeds up the processing.

4 Service Interfaces

For *LaaS* in conjunction with the macroscopic tool different request and response interfaces were agreed and realized. Based on XML descriptions following railML fragments, the different computational services calls are encrypted within a XML request document that is send to *LaaS*, implemented as a simple http-server listening on a predefined port. In general and for ongoing implementation activities the service interface will be realized as a full and stateless REST API. Information and computational service calls are distinguished by URL path extensions.

The requests to the services as well as the response generally consist of certain semantically separable sections. Basically, a human-readable format for requests and responses was intended. It has to be ensured, that references e.g. to infrastructure elements are consistent between caller and server.

4.1 Service Requests

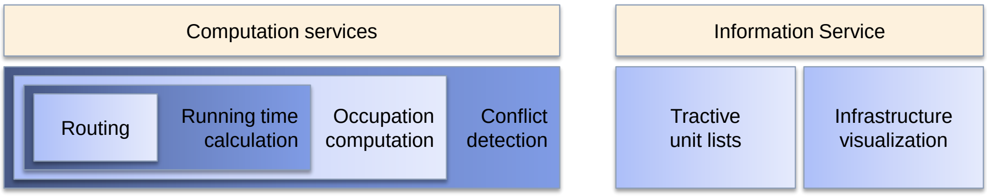

Figure 16: Service requests and service grouping

Service requests are specified within two groups (Figure 16): computation services and information services. The caller – especially the macroscopic tool using the provided services – has to collect and provide the request data and content.

The interesting requests are service requests, calling routing, running time calculation, occupation computation and conflict detection. The functionalities can be considered as a functional hierarchy: If occupation computation is requested, running time calculation and train routing is performed as preceding steps. Routing can be called as a base functionality but is always included in all other service calls, ensuring a consistent microscopic train route. If the specified service request always defines consistent train runs, the always executed routing functionality just acknowledges this.

An accepted service call example is shown in Figure 17.

```xml
<serviceCall service="fahrzeit">
    <description caption="Routing/FZR">Routing + FZR zwischen Berlin-Tief und Hamburg Hbf.</description>
    <trains>
        <train trainID="ICE 18/28_1518_BG1_DEFAUL" trainNumber="1518">
            <composition length="200" speed="300" weight="440" trainCategory="FRz">
                <trainControl lzb="true"/>
                <brakeUsage brakeType="compressedAir" airBrakeApplicationPosition="R" regularBrakePercentage="204"/>
                <vehicles>
                    <vehicle code="402-10.1" length="200" speed="300"/>
                </vehicles>
            </composition>
            <trainPath>
                <entry nodeID="BL"/>
                <entry nodeID="AH"/>
            </trainPath>
        </train>
    </trains>
</serviceCall>
```

Figure 17:Sample XML request to perform a route search and succeeding running time calculation for train 1518, ICE 402 traction unit, from Berlin (deep) to Hamburg Central. This information is available and maintained by macroscopic applications, too.

4.2 Service Responses

For computation service responses the response structure follows and duplicates the request XML structure, enriched by appropriate data extensions:

- For all requests the microscopic route-based routing information is added;
- For running time requests running times are added to the routing information;
- For occupation requests an extra list of occupation elements and occupation times is included;
- For conflict detection requests a further list is added to each train (if conflicting), describing occupation elements, opponent trains and conflict times and types.

The responses typically contain the request information and extend them by the microscopic data computed by *LaaS*.

An extract from the response to the routing and running time requests introduced before is shown in Figure 18 (note that the conflict service was queried to get occupation and conflict information).


```xml
<?xml version="1.0" standalone="yes"?>
<serviceCall service="konflikt">
    <trains>
        <train trainID="ICE 18/28_1518_BG1_DEFAUL" trainNumber="1518">
            <composition length="200" speed="300" weight="440">
                <trainControl lzb="true"/>
                <brakeUsage regularBrakePercentage="204" airBrakeApplicationPosition="R"/>
                <vehicles>
                    <vehicle code="402-10.1" speed="300" weight="440" length="200"/>
                </vehicles>
                <trainCategory>FRz</trainCategory>
            </composition>
            <trainPath>
                <entry elementID="839927" elementOcp="BL" elementType="track name"
                       elementPosition=" 1.700" elementName="25" enteringRoute="BPAF-U-6-U-BMOA"

                       leavingRoute="BPAF-U-6-U-BMOA"
                       stationTrack="6" halt="6"
                       arrival="00:00:38" departure="00:00:38"/>
                <entry elementID="833877" elementOcp="BMOA" elementType="reference point"
                       elementPosition=" 3.300" elementName="336" enteringRoute="BL-366/336/501-BWD"
                       leavingRoute="BL-366/336/501-BWD" stationTrack="336"
                       arrival="00:02:35" departure="00:02:35"/>
                  [...]
                <entry elementID="912330" elementOcp="AH" elementType="reference point"
                       elementPosition="286.663" elementName="305" enteringRoute="ABCH-305/556-AAR"
                       leavingRoute="ABCH-305/556-AAR"
                       stationTrack="305" halt="305" arrival="01:29:54" departure="01:29:54"/>
            </trainPath>
            <occupation>
                  <element from="00:00:21" until="00:00:59" type="occupation" elementID="839846"
                           elementOcp="BL" elementType="stopping position" elementPosition="2.211"
                           elementName="6/160"/>
                  <element from="00:00:21" until="00:00:59" type="occupation" elementID="839845"
                           elementOcp="BL" elementType="stopping position" elementPosition="2.185"
                           elementName="6/210"/>
                   [...]
            </occupation>
            <conflicts>
                  <conflict train1="1518" train2="1519" type="occupation (following)"
                            elementID="839846" elementOcp="BL"
                            elementType="stopping position" elementPosition=" 2.211"
                            elementName="6/160" from="00:00:21" until="00:00:39"/>
                  <conflict train1="1518" train2="2729" type="occupation (following)"
                            elementID="133891" elementOcp="HA"
                            elementType="stopping position" elementPosition=" 192.889"
                            elementName="6/210" from="01:28:11" until="01:28:45"/>
                   [...]
            </conflicts>
```

```
        </train>
    </trains>
</serviceCall>
```

Figure 18: Sample response containing the original request data as well as the derived routing information (“trainPath”) extended by times, occupation information related to infrastructure elements (“occupation”) and conflict information (“conflict”), also related to occupation elements.

5 Proof of Concept: LaaS – LUKS as a Service

Since 2017, *LaaS* was developed and realized as proof of concept; after first internal tests, a case study together with SMA und Partner AG, Zurich was set up to prove the suitability, feasibility and sustainability of the service oriented approach for external third party clients. During the case study, *LaaS* was improved and tailored as microscopic service for the macroscopic tool *Viriato*. Using this approach, the macroscopic tool was extended in a microscopic way and was finally enabled to compile timetables in an almost microscopic manner, seriously increasing the macroscopic planning quality with respect to feasible, conflict free timetables. The detailed consideration of station areas, switching zones and overlaps enriched the macroscopic planning quality by a yet unavailable and therefore undiscovered quality aspect.

During the realization, both parties focused on a smooth workflow, meaning that even advanced users of the macroscopic tool should not be hindered by the combination of two different programs. *LaaS* was almost transparently integrated into the user interface, providing microscopic routing services, running time calculation and conflict detection on demand. The proof of concept was performed within three steps:

a. Agreeing a proof scenario and a use-case to be evaluated: In most railway operations research projects, essential process steps are:
   1. Rough macroscopic planning of train runs to be considered,
   2. Transferring this first macroscopic timetable into the microscopic tool world,
   3. Microscopic conflict detection and, if necessary, modification of macroscopic timetable and looping back to step 2,
   4. Output of a macroscopic timetable that is conflict-free on microscopic level.

   In former joint projects, these iterations were identified to be very common and time consuming. In addition, proprietary interfaces were used, which could also lead to information loss. Target question is, if microscopic services can reduce this workload and source of errors.

b. Realizing service integration: The integration of *LaaS* into *Viriato* is incorporated into the macroscopic train editor. After defining the macroscopic train course, it is possible to send a request with the sequence of passed stations and train information to the *LaaS* service that answers by sending a response as introduced before.

Secondly, conflict detection visualization was realized and integrated into the macroscopic tool, showing and indicating the points of interest that have to be adopted within the macroscopic timetable.

c. Use-case evaluation: The evaluation of the use case succeeded:
   1. In contrast to former projects, results were generated much faster than before. The very first macroscopic planning steps had already a very high quality, and the above presented iteration can be shortened to one final microscopic check.
   2. Additionally, all process steps can be executed by a single user in a single tool using a common interface – there are fewer dependencies between team members with knowledge about different tools on the one side and proprietary interfaces on the other side.
   3. Performance has to be increased, but for this functional proof of concept this remains unconsidered. Due to the low frequency of service calls, this might moreover be acceptable considering the amount of workload to be saved.
   4. *LaaS* was very stable and requests were already answered meaningfully. In the seldom event of non-interpretable data even exceptions were reported with describing hints, the service reacted very robustly.

In general, from the *LaaS* point of view, the proof of concept confirmed the feasibility of advanced, sophisticated microscopic functionalities provided in a service oriented manner.

6 MaaS – Microscopy as a Service

The proof of concept – *LaaS* – described before was realized by extracting existing code fragments from LUKS® and providing special interfaces and functionalities in direct collaboration with the macroscopic caller. All calls, requests and response are specially designed and defined with respect to the intended proof scenario.

Beside using the now existing interaction for further projects carried out in direct joint-ventures or partnership, the selected and essentially studied approach of microscopic services and open interfaces will be extended, functionally enriched and made available under a more generic and universal perspective.

To underline this more universal intention to provide a wider variety of microscopic functionality and usability to a wider range of external users, scenarios and application fields the further development is continued as *Microscopy as a Service – MaaS*, a toolbox suite that is not only considered to be used "on demand" by a specific macroscopic tool, but also offers its wide-ranged microscopic computational power and expertise as services with specified, open communication. Currently intended extensions will focus e.g. on:

- The implementing of new infrastructure server interfaces,
- The support of multiple infrastructures and network layouts,
- Central storage and archiving services as well as
- Providing context sensitive service requests.

New services like layout and visualization services or simulation services are desirable and intended as well as an increase of scalability and performance and much more.

In the following, some few aspects of currently ongoing extensions and implementation activities are outlined. A first tool suite version is available by direct request.

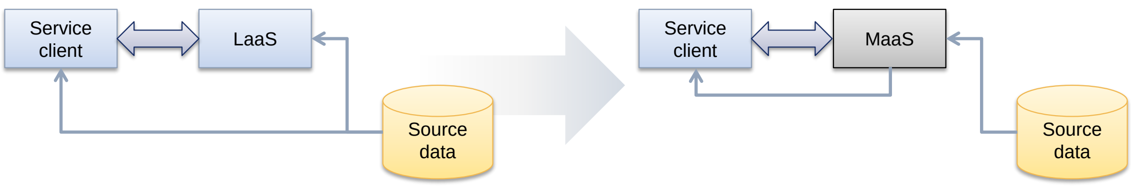

Figure 19: Improved infrastructure provision with MaaS

Infrastructure server interfaces: While one essential aspects of the presented proof of concept is the synchronized usage of source infrastructure data within all involved systems, *MaaS* will provide an interface for querying systems to request this information directly from *MaaS* services. With this step it can be ensured, that components querying *MaaS* have – at least – the chance to obtain the same infrastructural information as the microscopic service they contact (Figure 19). In this way, a "single point of truth" with respect to infrastructure information can be realized while with the current proof of concept, data compatibility between the tools had to be ensured manually on behalf of using the same input data sets.

Multiple infrastructures and network layouts: Another extension with focus on a practical usability is the handling of different infrastructures that might be used for different infrastructure layouts, studies or alternative evaluations (e.g. closures or limitations due to construction works etc.).

In that way, *MaaS* can be used for a wide variety of users and probably opened for different external consultants, clients and departments within a company using *MaaS*.

Context sensitive services: Maybe one of the most interesting extensions with respect to component oriented services within timetabling activities will be the ability to define context sensitive data interactions.

It will e.g. be possible to define train runs, request running times and occupation computations as already existing, but additionally persisting information for succeeding requests or to withdraw them later. In that way, the sequence of requests can set up a context of information – the timetable – that represent the collected and aggregated knowledge at a time, allowing e.g. the evaluation of conflicts at a certain time on behalf of this former data collection. In that way, the GUI of microscopic timetabling systems can be separated from the microscopic timetabling functionality and e.g. provided by third parties or customized as required.

7 Conclusion

With *LaaS* the proof of concept succeeded, that microscopic timetabling functionalities can be extracted, modularized and provided as services for a beneficial usage by macroscopic tools to optimize work processes and to increase timetable planning quality seriously.

The experiences made within this project confirm the concept of providing microscopic functionalities "out of a box" and motivate to assemble the comprehensive microscopic commercial toolbox suite *MaaS* publishing some very sophisticated technics from microscopic approaches in a very simple, easily usable way.

Keywords:

Railway operations research, microscopic data modelling, service oriented architecture, modularization, centralized services, software as a service

Topic classification:

Innovative Eisenbahnbetriebskonzepte: Servicekonzepte und Praxispaper

Authors:

Dr. Kuckelberg, Alexander
VIA Consulting & Development GmbH
Römerstr. 50, 52064 Aachen
a.kuckelberg@via-con.de, +49 241 436662-16

Janecek, David
VIA Consulting & Development GmbH
Römerstr. 50, 52064 Aachen
d.janecek@via-con.de, +49 241 436662-19