A SOFTWARE ARCHITECTURE FOR AUTONOMOUS MAINTENANCE SCHEDULING: SCENARIOS FOR UK AND EUROPEAN RAIL

CHRIS TURNER[1], PRITHYUKSHAA THOPPUR RAVI[1], ASHUTOSH TIWARI[1], ANDREW STARR[1], & KEVIN BLACKTOP[2]
[1]Manufacturing and Materials Department, Cranfield University, UK.
[2]Network Rail, The Quadrant, UK.

ABSTRACT
A new era of automation in rail has begun offering developments in the operation and maintenance of industry standard systems. This article documents the development of an architecture and range of scenarios for an autonomous system for rail maintenance planning and scheduling. The Unified Modelling Language (UML) has been utilized to visualize and validate the design of the prototype. A model for information exchange between prototype components and related maintenance planning systems is proposed in this article. Putting forward an architecture and set of usage mode scenarios for the proposed system, this article outlines and validates a viable platform for autonomous planning and scheduling in rail.
Keywords: decision support systems, rail planning and scheduling, software architecture.

1 INTRODUCTION

According to the Office of Rail Regulation (ORR) [1], for the period 2013–14 total rail industry expenditure in the UK was £12.7 bn, of which £6.2 bn (49%) was incurred in operating rail infrastructure alone. A significant percentage of these costs were incurred through maintenance of railway infrastructure. Railway infrastructure state may be affected by a range of factors such as 'the track geometry, topography, geology and weather conditions' [2]. Maintenance plans must be devised and the risk of failures forecasted in order to maintain safe operation with high levels of service availability.

With a significant increase in railway traffic in the UK expected over the coming years the scheduling and distribution of maintenance (and possession) time is increasingly challenging. Hence 'An integrated maintenance software framework' is essential to streamline future maintenance activities based on factors of cost, urgency and the flagging of issues sensed by intelligent assets and infrastructure. These factors, especially the rise in the use of sensors, are driving a need for the automated planning of maintenance tasks within rail. With this need comes the potential for autonomous scheduling.

1.1 The AUTONOM project

The examination of autonomous planning and scheduling for rail maintenance is a core component of the AUTONOM project (integrated through life support for high-value systems) [3] sponsored by the Engineering and Physical Sciences Research Council (EPSRC), a UK government–backed major research funder and UK rail infrastructure provider Network Rail

This paper is part of the proceedings of the 15th International Conference on Railway Engineering Design and Operation (COMPRAIL)
www.witconferences.com

ISSN: 2058-8305 (paper format), ISSN: 2058-8313 (online), http://www.witpress.com/journals
DOI: 10.2495/TDI-V1-N3-371-381

(Network Rail is the organization that is responsible for maintaining and developing the UK rail infrastructure including signalling, bridges, tunnels, level crossings, viaducts and 17 key stations within the country) [4]. Encompassing areas such as sensor fusion (collection and analysis of data streams from rail vehicles and infrastructure) and cost analysis (costs and benefits in maintenance decisions) the development of a detailed architecture, encompassing scenarios of operation, was necessary. The AUTONOM system is comprised of four modules:

- WP1: Integration – integrates system functionality and acts as a user 'dashboard'
- WP2: Sensor Fusion – gathers data from data stores and live feeds, provides an analysis of asset degradation trends and alerts to flag areas in need of urgent maintenance
- WP3: Planning and Scheduling – optimally schedules maintenance jobs based on importance and cost
- WP4: Costing – determines the cost and benefit of different maintenance jobs

While other notation sets such as Business Process Modelling Notation (BPMN) and Flowcharts [5] have been used in enterprise level projects, the Unified Modelling Language (UML) is still the accepted modelling standard. Examples of UML use in rail are in evidence through works such as Berkenkotter *et al*. [6, 7]. The modelling standard provides a set of notations designed to support various domain specialisms and stages involved in the engineering of software systems.

2 RELEVANT RESEARCH

Planning in railway operations is a complex task, due to the large solution space this work is normally partitioned into several problems that are solved sequentially [8]. Klabes [9] presents a novel framework detailing the different planning processes involved:

- Network planning: determines the detailed layout of the railway infrastructure
- Line planning: determines the lines and the frequency of train operation on them
- Timetabling: governs the arrival and departure times of trains at train stations
- Capacity allocation: including insertion of train path requests into the working train timetable
- Vehicle planning: rolling stock assignment
- Crew planning: crew rostering and staff planning
- Re-scheduling: controlling the movements of trains during operation

To date, many studies have examined the problem of maintenance scheduling for railway systems. The maintenance concept can be categorized into three groups: corrective maintenance; periodic maintenance; predictive maintenance [10]. Though works on autonomous operation and scheduling in rail are limited, one particular study by Dadashi et al. [11] suggests an intelligent infrastructure to move from find and fix to predict and prevent. Schlake *et al*. [12] have conducted research into the trackside monitoring of rail vehicles, an essential development to enable predictive maintenance. Their work [12] examined the economic impact of train delays and the effect of introducing lean production methods in passenger and freight railway operations to improve rail vehicle maintenance and monitoring procedures.

Automated planners have been investigated by authors such as Cresswell *et al*. [13] and Fernandez *et al*. [14]. Such planners require action models described using languages such as the Planning Domain Definition Language (PDDL) [15].

Apart from such standardized data formats, a semantic model with a high level of 'structured interoperability' between information systems is required to correctly combine and

manage complex scheduling information. Verstichel *et al*. [16] describe both UML and OWL as methods of semantically describing models. Marcano *et al*. [17] while stating that UML offers a standard systems modelling notation also acknowledge that it lacks formal semantics to model safety critical scenarios.

In summary, the literature suggests that interactions and message passing are modelled either using sequence diagrams or interaction diagrams, the former being more prevalent. As far as message exchange is concerned, numerous research articles point towards an XML-based schema due to its simplicity and dual advantage of being both machine and human readable.

3 METHODOLOGY

The methodology followed for this research is detailed in Fig. 1. As shown in Fig. 1 the background study phase consisted of two sub-tasks, namely to study relevant literature for modelling in the context of railways and to analyse the input and outputs between the systems/modules within the maintenance framework. The literature review performed helped to identify the UML approaches present. The model and document phases dealt with modelling the interactions and message passing between the aforementioned modules. UML was identified from literature to be the most suitable visual modelling language for the case study. It was decided at this stage that the standard notation set of UML should be used in this research due to its familiarity and acceptability to both software developers and the rail industry (with the proviso that such models could be transformed to newer more rail specific standards, as they gain in popularity and use, in the future). The validation phase involved identifying operational scenarios from domain experts, pertaining to this modelling environment.

4 SCENARIOS FOR AUTONOMOUS PLANNING AND SCHEDULING

As previously mentioned in this article the use of UML as a notation for the description of software including complex railway information systems is quite established, it is by far the most widely accepted modelling language in the software engineering community. In the development of the scenario set supporting the AUTONOM project the process of modelling is divided into three phases, namely scenario specification phase, architecture engineering phase and interaction design phase.

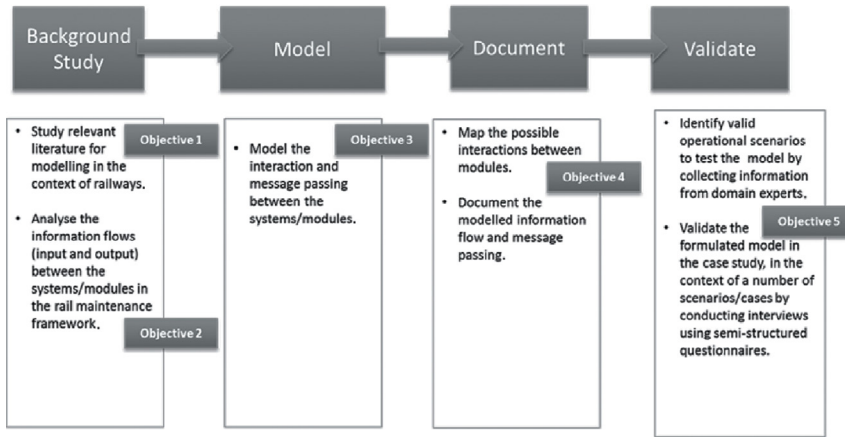

Figure 1: Methodology employed.

4.1 Scenario descriptions

In order to completely specify and model the requirements of the AUTONOM system, it is essential to understand the scenarios that it can work under. With careful analysis and discussions with the stakeholders, the following scenarios were identified:

a. amanual production of maintenance schedules
b. bsemi-automated production of maintenance schedules
c. cautomated production of maintenance schedules
d. dautomated/person-in-loop production of maintenance schedules

Scenario 1: Manual production of maintenance schedules
The first scenario deals with the production of schedules for further processing from existing Network Rail systems; a 'person in the loop' manually prompts AUTONOM to start running. In this scenario, the data is static in nature and comes from a database store at Network Rail (NR). AUTONOM needs to identify manual alterations and produce schedules that will be stored in the database shared with all NR's systems. The decision dashboard is generated and displayed to user. The scenario ends with user having to manually select one of the many scheduling choices from the dashboard.
Scenario 2: Semi-automated production of maintenance schedules
The semi-automated use of AUTONOM involves the generation of automatic alerts to trigger actions within the system. The alert is generated after studying the live data streams supplied by NR systems. Communication protocols at both NR and AUTONOM ends need to be devised to respond to such alerts. As with Scenario 1 the ultimate decision of schedule selection is manual.
Scenario 3: Automated production of maintenance schedules
In this scenario a user is not required as an autonomous decision engine is used to select a schedule as against manual selection in Scenario 2. In this scenario the system is fully autonomous, notifying NR systems of updated maintenance schedules when available, based on alerts raised by the sensor fusion module.
Scenario 4: Automated/person-in-loop production of maintenance schedules
Automated/person-in-loop is an extension of Scenario 3 where a user can veto autonomous decisions proposed by the system; abnormal values and conditions are then monitored and communicated to the user via the dashboard. When such situations occur, an alert is generated which requires user intervention for a decision to be made. The user can either continue with the scheduling advised by AUTONOM or abort the suggested flow of processes and take over manual control of the operations.

4.2 Scenario 1: Manual production of maintenance schedules

There are two actors in this use case, namely User and the AUTONOM system itself. The user clicks on the dashboard to update the workflow log. This workflow log is used to identify different activities that are performed as a part of the maintenance process. The AUTONOM system reads the static data, listens for alterations in state (raised by the user starting the system), generates the decision dashboard, creates schedules and parses the workflow log. In addition a user may wish to manually import data from a live Hadoop big data store.

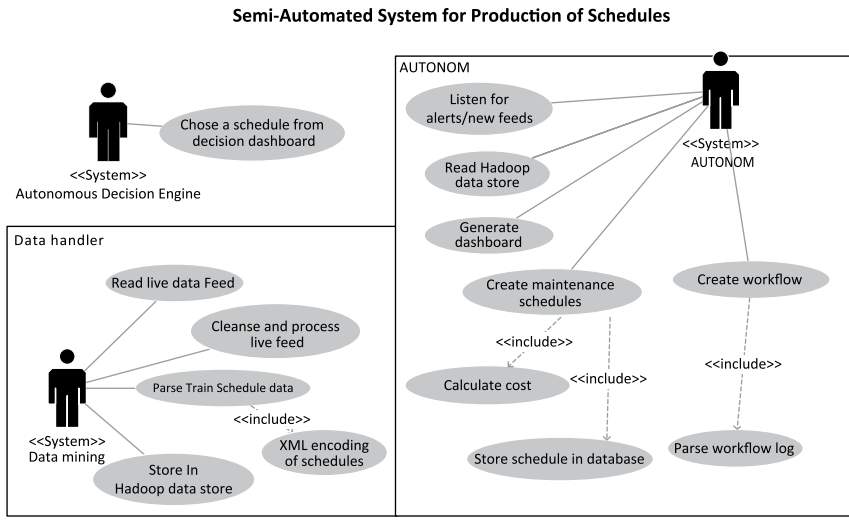

Figure 2: Use case diagram for semi-automated system for production of schedules.

4.3 Scenario 2: semi-automated

There are three actors in this use case. The additional actor is provided by the 'data handler' 'live data streams'. In this scenario the user has one task, that of selection of a schedule from the decision dashboard returned by AUTONOM. In this scenario AUTONOM listens for alterations that can be triggered by an incoming data feed as against the manual mode in Fig. 2. The processes such as 'Generate Dashboard' and 'Create Schedules' are similar to the manual mode of operation. In this scenario the Hadoop big data store is utilized. Creation of workflow including parsing is performed at the AUTONOM end. The data handler reads the live data feeds. It also creates an XML encoding of the existing schedules and stores both the schedules and cleansed data in the Hadoop data store.

4.4 Scenario 3: automated

The use case diagram (shown in Fig. 3) details the system working in fully autonomous mode. In Fig. 3 it can be seen that the user has been removed from the loop.

4.5 Scenario 4: automated /person-in-loop

In this scenario the user is placed back into the diagram for this scenario. The actors AUTONOM system, data handler and autonomous decision engine execute the same set of processes as found in Fig. 3. The autonomous decision engine apart from choosing a schedule also checks for abnormalities and notifies the user of an unexpected event. The user is simply tasked to act on such alerts by approving the course of action suggested by AUTONOM.

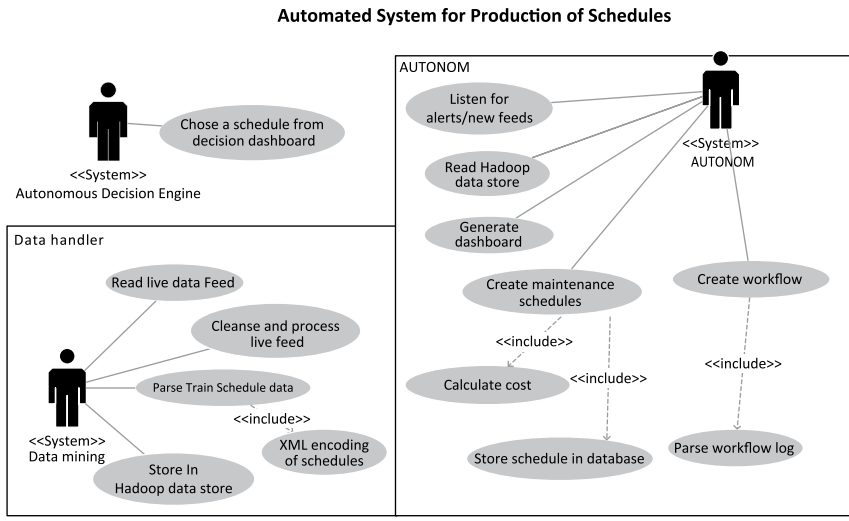

Figure 3: Use case for automated system for production of schedules.

4.6 Sequence diagrams of the use cases

The case scenarios can be modelled as a sequence diagram depicting the messages exchanged as in Appendix 1 (showing the sequence for the automated scenario). The initial objects involved are the user, AUTONOM prototype, static data store, the database (to store the schedules), NR systems and a 'workflow handler' which is again a part of AUTONOM. An additional component in this diagram is the Hadoop data store (in the manual scenario it is an inactive entity due to the use of static data). With the fully automated production of schedules, the AUTONOM prototype along with NR systems listens for alerts (or new data feeds). Meanwhile the 'live data handler' works with the live data streams, cleanses live data and parses the data on schedule (to represent schedules as XML). Once the data is stored in the Hadoop data store by 'live data handler', a notification is received by the AUTONOM software of an incoming feed. The production of maintenance schedules is commenced by reading the Hadoop data store. A set of schedules and dashboard of possible schedules are produced.

In the fully autonomous scenario the user remains inactive until and unless the 'autonomous decision engine' creates an alert for the user to handle an emergency situation.

4.7 Sequence diagram for whole AUTONOM project

The next step is to model the information exchange between the AUTONOM modules described in Section 1.1, this is shown in Fig. 4. Apart from these four main components, there exists another object: the Web Application. This application is used to mirror the dashboard displayed on desktop application through a web browser, thereby enabling portability across several devices.

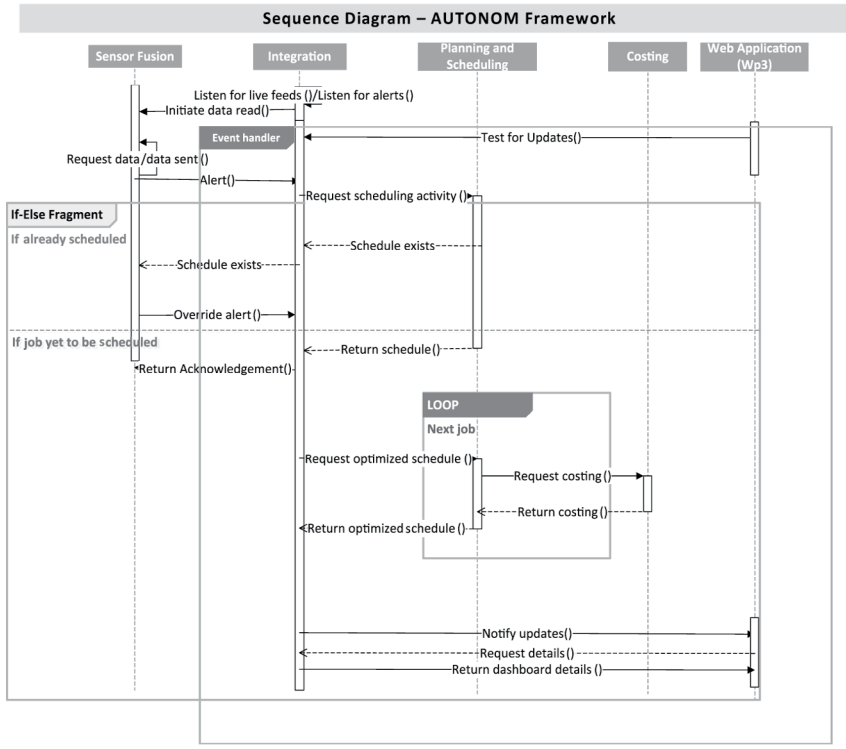

Figure 4: Sequence diagram for the AUTONOM framework.

There are two UML frame elements being used in this sequence diagram: one to denote an alternative fragment and another to denote a loop fragment. The process begins with the 'integration module' listening for alerts or new feeds. When a new data stream is detected, the integration module directs the sensor fusion module to read data from the corresponding data store (could be static data or a live feed via the Hadoop data store). The data is read and processed by the sensor fusion work package; it then generates an alert when the data values display the presence of a fault. The sensor fusion module creates an alert and directs integration to schedule a maintenance activity to manage the fault detected. The integration module subsequently requests the scheduling of the activity. The planning and scheduling module either returns a schedule or messages integration about the existence of that particular activity in the schedule. In the case of the schedule already being present, the sensor fusion overrides the alert. Alternatively, the integration module sends an acknowledgement to the senor fusion module. The model element 'Alternative Segment' is used in order to visually represent the two aforementioned mutually exclusive logic flows. Once the planning and scheduling module receives a request to produce an optimized schedule, it triggers the costing module to calculate the schedule's cost. This action is completed for numerous combinations of schedules, for each of which the

cost value is obtained from the costing module. This iterative fragment is denoted by a 'loop' segment in the diagram. The planning and scheduling module, finally, passes the optimized schedule to the integration module dashboard. The Web Application mirrors this action.

5 VALIDATION

In the validation of the architecture a semi-structured questionnaire was drafted and the insights of eight industry experts sought. This questionnaire was used to judge the models based on various parameters such as completeness, simplicity, clarity and accuracy. The user community chosen for this validation process were industry experts with experience in planning and scheduling and the design and development of prototypes. The questionnaire for this study consisted of seven parts, beginning with an introduction a second section stated details of the case study, and the subsequent sections listed: UML models to be evaluated; questions on scenario modelling; use case diagrams; AUTONOM component interaction; sequence diagrams. The message passing aspect was tested using questions under Section 6 followed by questions about the overall model. For each of the questions, the experts were expected to rate the model (from a rating scale of 1, low, to 5, high). The overall feedback from the eight interviews conducted showed parameters such as accuracy, simplicity and completeness scored more than 4.5. The clarity of the models was evaluated to be 4.3 out of 5.

The main changes made in response to expert comments included the insertion of a sub-task within `create maintenance schedules', called `calculate cost', in the use case diagrams and improvements in the way the data store was represented in the sequence diagrams. The data stores are presented as hollow rectangles whereas the elements representing functions are shown as solid coloured figures.

6 DISCUSSION AND CONCLUSIONS

This article has now presented the UML modelling pertaining to AUTONOM. The study includes the high level design (HLD) of the prototype with this article focussing on the communication between identified modules in the AUTONOM framework. Scenario-based modelling best describes the information flow between the different modules. UML provides a unique bridge for developers and business users, permitting efficient communication that supports the implementation of software. In the further development of this research the following suggestions for future work need to be taken into account: An XML-based message framework should be employed to specify generic templates of various messages that could be exchanged. Description of constraints is an essential step in introducing railway terminology into the UML profile. UML standard supports a notation, the Object Constraint Language (OCL) for specification of constraints. In this study, scenarios were used as the foundation for requirement specification. Use case models derived from the identified scenarios project a black box representation of the system. The use of sequence diagrams with use cases allow for clarity in software functionality. Together, this set of UML diagrams will act as a standardized documentation of the system and will be used to communicate system specifications to the implementation team.

REFERENCES

[1] GB rail industry financial information 2013–14 (Office of Rail Regulation), available at http://orr.gov.uk/__data/assets/pdf_file/0005/16997/gb-rail-industry-financials-2013-14.pdf (accessed 4 February 2016).

[2] Cores, F., Caceres, N., Benitez, F. G., Escriba, S. & Jimenez Redondo, N., A logical framework and integrated architecture for the rail maintenance automation. In European Transport Conference 2013, 2013/9/30 to 2013/10/2, Frankfurt.

[3] AUTONOM (integrated through life support for high-value systems), available at https://www.cranfield.ac.uk/research/research-activity/current-projects/research-projects/autonom.html (accessed 4 February 2016).

[4] Turner, C., Tiwari, A., Starr, A. & Blacktop, K., A review of key planning and scheduling in the rail industry in Europe and UK. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, **230**(3), 2015. DOI: 10.1177/0954409714565654.

[5] Mary, S.R. & Rodrigues, P., Software architecture-evolution and evaluation. *International Journal of Advanced Computer Science and Applications* **3**(8), pp. 82–88, 2012. DOI: 10.14569/IJACSA.2012.030814.

[6] Berkenkötter, K., Bisanz, S., Hannemann, U. & Peleska, J., Executable hybrid UML and its application to train control systems. *Integration of Software Specification Techniques for Applications in Engineering,* eds. H. Ehrig, W. Damm, M. Große-Rhode, W. Reif, E. Schnieder & Westkamper, Springer Verlag: Berlin, Germany, pp. 145–173, 2004.

[7] Berkenkötter, K. & Hannemann, U., Modeling the railway control domain rigorously with a uml 2.0 profile. *Computer Safety, Reliability, and Security (LNCS – 4166)*, ed. J. Górski, Springer, Heidelberg: Springer, pp. 398–411, 2006.

[8] Caprara, A., Kroon, L., Monaci, M., Peeters, M. & Toth, P., Passenger railway optimization. *Handbooks in Operations Research and Management Science* **14**, pp. 129–187, 2007. DOI: 10.1016/S0927-0507(06)14003-7.

[9] Klabes, S.G., Algorithmic railway capacity allocation in a competitive European railway market, Doctoral dissertation, PhD thesis, RWTH Aachen, 2010.

[10] Camci, F. & Chinnam, R.B., *Process Monitoring, Diagnostics and Prognostics in Machining Processes*, Saarbrücken, Germany: LAP Lambert Academic Publishing, 2010.

[11] Dadashi, N., Wilson, J.R., Sharples, S., Golightly, D. & Clarke, T., A framework of data processing for decision making in railway intelligent infrastructure. *2011 IEEE First International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, Miami Beach, FL, IEEE, pp. 276–283, 2011.

[12] Schlake, B.W., Barkan, C.P.L. & Riley Edwards, J., Train delay and economic impact of in-service failures of railroad rolling stock. *Journal of the Transportation Research Board*, **2261**, pp. 124–133, 2011. DOI: 10.3141/2261-14.

[13] Cresswell, S.N., McCluskey, T.L. & West, M.M., Acquiring planning domain models using LOCM. *Knowledge Engineering Review*, **28**, pp. 195–213, 2013. DOI: 10.1017/S0269888912000422.

[14] Fernández, S., De La Rosa, T., Fernández, F., Suárez, R., Ortiz, J., Borrajo, D. & Manzano, D., Using automated planning for improving data mining processes. *Knowledge Engineering Review*, **28**, pp. 157–173, 2013. DOI: 10.1017/S0269888912000409.

[15] McDermott, D., The AIPS'98 Planning Competition Committee. PDDL–the planning domain definition language. Tech. rep., available at http://www.cs.yale.edu/homes/dvm/ (accessed 18 September 2015).

[16] Verstichel, S., Ongenae, F., Loeve, L., Vermeulen, F., Dings, P., Dhoedt, B. & De Turck, F., Efficient data integration in the railway domain through an ontology-based methodology. *Transportation Research Part C: Emerging Technologies*, **19(4)**, pp. 617–643, 2011. DOI: 10.1016/j.trc.2010.10.003.

[17] Marcano, R. & Levy, N., Using B formal specifications for analysis and verification of UML/OCL models. In Workshop on consistency problems in UML-based software development. *5th International Conference on the Unified Modeling Language*, Dresden, Germany, pp. 91–105, 2002.

APPENDIX 1: SEQUENCE DIAGRAM FOR FULLY AUTOMATED PRODUCTION OF SCHEDULES