

# Na tomto místě bude oficiální zadání vaší práce

- Toto zadání je podepsané děkanem a vedoucím katedry,
- musíte si ho vyzvednout na studijním oddělení Katedry počítačů na Karlově náměstí,
- v jedné odevzdané práci bude originál tohoto zadání (originál zůstává po obhajobě na katedře),
- ve druhé bude na stejném místě neověřená kopie tohoto dokumentu (tato se vám vrátí po obhajobě).



Czech Technical University in Prague  
Faculty of Electrical Engineering  
Department of Computer Science and Engineering



Bachelor's Project

**Tool for production processes operational monitoring**

*Luis Moreno*

Supervisor: Martin Klíma Ing., Ph.D.

Study Programme: Softwarové technologie a management, Bakalářský

Field of Study: Softwarové inženýrství

May 23, 2014



## Acknowledgements

I would like to express my gratitude to my thesis supervisor Martin Klíma Ing., Ph.D. for his patient guidance, enthusiastic encouragement and constructive criticism. I would also like to thank my colleague Ondřej Harcuba for his advices and close cooperation during the development.

Finally, I wish to thank my mother and my girlfriend for their support and encouragement throughout my study.



## Declaration

I hereby declare that I have completed this thesis independently and that I have listed all the literature and publications used.

I have no objection to usage of this work in compliance with the act §60 Zákon č. 121/2000Sb. (copyright law), and with the rights connected with the copyright act including the changes in the act.

In Prague on May 22, 2014

.....





# Abstract

This thesis describes the development of the Scenario designer: a software tool developed within the European project ARUM.

The tool aims to provide a graphical user interface that allows its user to set up, visualize and monitor ramp-up productions. The thesis contains the functional requirements analysis, architecture, design and the partial implementation of the Scenario designer (SD) graphical interface. The SD has been developed as a part of the Factory network&Scenario designer tool (FNSD)[6]. Effort was made to follow user-centered design processes during the development of both tools and a sub-set of the FNSD interface design was tested with end users. The obtained feedback resulted in the design adjustment according to the user's needs and preferences.

The result of this work is the design of a managerial tool to support ramp-up productions, developed according to gathered requirements from earlier ARUM project phases.

# Abstrakt

Tato práce se zabývá vývojem softwarového nástroje Scenario designer: aplikace vyvíjená v rámci běžícího evropského projektu ARUM. Cílem práce je vyvinout manažerský nástroj pro konfiguraci, monitorování, vizualizaci a rozvrhování výroby v ramp-up prostředí.

Tento dokument obsahuje analýzu funkčních požadavků, návrh architektury, návrh uživatelského rozhraní a popis jeho dosavadní implementace. Scenario designer je vyvíjen jako součást aplikace Factory network&Scenario designer (FNSD)[6]. Od počátku vývoje bylo snahou dodržet principy tzv. "user-centered" designu. Část navrženého uživatelského rozhraní FNSD byla testována s koncovými uživateli a získaná zpětná vazba vedla k úpravě a přizpůsobení nástroje jejich reálným potřebám.

Výsledkem práce je návrh uživatelského rozhraní, vyvinuté v souladu s požadavky projektu ARUM za účelem podpory výroby v ramp-up prostředí.



# Contents

.....	ix
<b>1 Introduction</b>	<b>1</b>
1.1 ARUM project	1
1.2 Objective of the thesis	2
.....	2
1.2.1 Former task	2
1.2.2 Latest work	3
.....	3
.....	3
1.3 Document outline	3
<b>2 Analysis</b>	<b>5</b>
2.1 Sources of information	5
.....	5
.....	5
2.2 Domain description	5
.....	6
2.2.1 Ramp-up	6
.....	6
.....	6
2.2.2 Needs in production management	6
2.2.3 ARUM solution	7
.....	7
.....	7
.....	7
2.3 Refinement of requirements	7
2.3.1 Specification of user roles	8
2.3.2 Functional requirements	8
2.3.2.1 Functional requirement 1.1	9
2.3.2.2 Functional requirement 1.2	9
2.3.2.3 Functional requirement 1.3	9
2.3.2.4 Functional requirement 1.4	10
2.3.2.5 Functional requirement 1.5	10
2.3.2.6 Functional requirement 1.6	10
2.3.2.7 Functional requirement 1.7	11

2.3.2.8	Functional requirement 1.8	11
2.3.2.9	Functional requirement 1.9	11
2.4	Functional specification	13
2.4.1	Scene life cycle	13
	. . . . .	14
2.4.2	Load/create scene	15
2.4.3	Load scene(s)	16
2.4.4	Edit scene	16
2.4.5	Add station	17
2.4.6	Edit scene properties	18
2.4.7	Setup events	18
2.4.8	Add event	19
2.4.9	Set up station	20
2.4.10	Schedule	21
2.4.11	Result analysis	21
<b>3</b>	<b>Design and Implementation</b>	<b>23</b>
3.1	Application description	23
3.2	Architecture of the FNSD	25
3.2.1	ARUM system architecture diagram	25
3.2.2	System architecture components	26
3.2.3	Communication within the ESB	27
	. . . . .	27
	. . . . .	27
3.2.3.1	Errors handling	27
3.2.4	Load scene	28
3.2.5	Create scene	29
3.2.6	Update scene	30
3.2.7	Delete scene	31
3.2.8	Scheduling	32
3.2.9	Scheduling involving MIDAS	34
3.3	Scenario designer graphical interface	36
	. . . . .	36
3.3.1	FNSD components and layout	36
3.3.1.1	Login window	36
3.3.1.2	Main window	37
	. . . . .	38
	. . . . .	38
3.3.1.3	Sub windows	38
3.3.1.4	Quick access tool-bar	39
	. . . . .	39
3.3.1.5	Ribbon	40
	. . . . .	40
	. . . . .	40
	. . . . .	40
	. . . . .	41

3.3.1.6	Main content panel . . . . .	41
	. . . . .	42
3.3.1.7	Browser panel . . . . .	43
	. . . . .	43
	. . . . .	43
	. . . . .	43
3.3.1.8	Properties panel . . . . .	44
	. . . . .	44
	. . . . .	45
3.3.1.9	Navigation panel . . . . .	45
	. . . . .	45
	. . . . .	45
3.3.1.10	Search component . . . . .	46
	. . . . .	46
	. . . . .	46
	. . . . .	46
3.3.1.11	Tables . . . . .	46
	. . . . .	47
	. . . . .	47
3.3.1.12	File chooser . . . . .	47
	. . . . .	48
3.3.2	Scenario designer components . . . . .	48
3.3.2.1	Scene management . . . . .	48
	. . . . .	48
	. . . . .	48
3.3.2.2	Scenario designer ribbon . . . . .	49
	. . . . .	49
	. . . . .	50
3.3.2.3	Scenes and simulations browser . . . . .	51
3.3.2.4	Scenario designer properties and navigation panel . . . . .	52
	. . . . .	52
3.3.2.5	Station overview . . . . .	53
3.3.2.6	Add stations from multiple scenes . . . . .	54
	. . . . .	54
3.3.2.7	Add events . . . . .	55
	. . . . .	55
	. . . . .	55
	. . . . .	55
3.3.2.8	New scene . . . . .	56
3.3.2.9	Station setup wizard - general properties . . . . .	56
3.3.2.10	Station setup wizard - assembly plans assignment . . . . .	57
3.3.2.11	Station setup wizard – human resources setup . . . . .	58
	. . . . .	58
	. . . . .	58
	. . . . .	58
	. . . . .	58

3.3.2.12	Station setup wizard - tools setup . . . . .	59
3.3.2.13	Station setup wizard - tool catalogue . . . . .	61
3.3.2.14	Scheduling properties setup . . . . .	61
3.3.2.15	KPIs comparison . . . . .	63
	. . . . .	63
3.3.2.16	Schedules graphical comparison . . . . .	64
	. . . . .	64
3.3.3	Production mode . . . . .	64
3.3.3.1	Running production monitoring - stations overview . . . . .	65
3.3.3.2	Running production monitoring - dashboard . . . . .	65
	. . . . .	65
3.4	Technology used for the implementation . . . . .	67
	. . . . .	67
3.4.1	JavaFX . . . . .	67
3.5	High fidelity prototypes implementation . . . . .	68
	. . . . .	68
	. . . . .	68
3.5.1	Scene management . . . . .	68
3.5.2	New scene . . . . .	69
3.5.3	Station setup wizard - human resources setup . . . . .	69
<b>4</b>	<b>Usability testing</b>	<b>73</b>
4.1	Usability testing . . . . .	73
4.1.1	Purpose of the usability testing and expected output . . . . .	73
	. . . . .	74
4.1.2	Scheduling properties . . . . .	74
4.1.2.1	Issue: Redundant setup – preferred optimization . . . . .	74
4.1.2.2	Issue: Scheduling time range – current cycle as default . . . . .	74
4.1.2.3	Issue: Scheduling properties – swapped areas . . . . .	75
4.1.3	Results comparison . . . . .	75
4.1.3.1	Issue: Erroneous industrial minutes abbreviation . . . . .	75
4.1.3.2	Issue: Add new KPI – amount of travelling work . . . . .	75
4.1.3.3	Issue: Change throughput unit . . . . .	76
<b>5</b>	<b>Conclusion</b>	<b>77</b>
5.1	Thesis summary . . . . .	77
	. . . . .	77
5.1.1	Further work . . . . .	77
<b>A</b>	<b>Glossary</b>	<b>81</b>
<b>A</b>	<b>Content of attached CD</b>	<b>85</b>

<b>A Usability testing scenarios</b>	<b>87</b>
A.1 Rescheduling scenario . . . . .	87
. . . . .	87
. . . . .	87
. . . . .	88
. . . . .	88
. . . . .	88





# List of Figures

2.1	Scene life cycle sequence diagram . . . . .	14
2.2	Scene lifecycle process . . . . .	15
2.3	Load/create scene process . . . . .	15
2.4	Load scene(s) process . . . . .	16
2.5	Edit scene sub-process . . . . .	17
2.6	Add station sub-process . . . . .	17
2.7	Edit scene properties sub-process . . . . .	18
2.8	Setup events sub-process . . . . .	19
2.9	Add event sub-process . . . . .	19
2.10	Set up station sub-process . . . . .	20
2.11	Schedule sub-process . . . . .	21
2.12	Result analysis sub process . . . . .	21
3.1	ARUM system architecture . . . . .	25
3.2	Load scene . . . . .	28
3.3	Create scene . . . . .	29
3.4	Update scene . . . . .	30
3.5	Delete scene . . . . .	31
3.6	Scheduling . . . . .	32
3.7	Scheduling involving MIDAS . . . . .	34
3.8	FNSD login window . . . . .	37
3.9	FNSD main application window . . . . .	38
3.10	FNSD modal window . . . . .	39
3.11	FNSD quick access tool-bar . . . . .	40
3.12	FNSD ribbon - mode switch . . . . .	41
3.13	FNSD Main content panel . . . . .	43
3.14	FNSD Browser panel . . . . .	44
3.15	FNSD Properties panel . . . . .	44
3.16	FNSD Navigation panel . . . . .	45
3.17	FNSD Search component . . . . .	46
3.18	FNSD Table example . . . . .	47
3.19	FNSD File chooser example . . . . .	48
3.20	Scene management - AIB . . . . .	49
3.21	Scenario designer ribbon . . . . .	50
3.22	Scenario designer browser panel - scenes tab . . . . .	51

3.23	Scenario designer browser panel - simulations tab . . . . .	52
3.24	Scenario designer properties panel . . . . .	52
3.25	Scenario designer navigation panel . . . . .	53
3.26	Station overview . . . . .	53
3.27	Add stations from multiple scenes . . . . .	54
3.28	Add events . . . . .	55
3.29	Scenario designer - new scene setup . . . . .	56
3.30	Station setup wizard - general properties . . . . .	57
3.31	Station setup wizard - assembly plans assignment . . . . .	57
3.32	Station setup wizard – Human resources setup . . . . .	59
3.33	Station setup wizard - tools setup . . . . .	60
3.34	Station setup wizard - tool catalogue . . . . .	61
3.35	Scheduling properties setup . . . . .	62
3.36	Scenario designer - KPIs comparison . . . . .	63
3.37	Scenario designer - KPIs comparison . . . . .	64
3.38	Production mode - stations overview . . . . .	65
3.39	Production mode - dashboard . . . . .	66
3.40	SD high fidelity prototype - scene management . . . . .	69
3.41	SD high fidelity prototype - scene management . . . . .	70
3.42	SD high fidelity prototype - scene management . . . . .	71
4.1	Rescheduling - reported issues . . . . .	74
4.2	Results comparison - reported issues . . . . .	75

# List of Tables

2.1	User roles mapping . . . . .	8
3.1	Load scene data flow methods . . . . .	28
3.2	Create scene data flow methods . . . . .	29
3.3	Delete scene data flow methods . . . . .	31
3.4	Update scene data flow methods . . . . .	32
3.5	Scheduling data flow methods . . . . .	33
3.6	Scheduling involving MIDAS data flow methods . . . . .	35
3.7	User roles to application modes mapping . . . . .	37
A.1	Glossary . . . . .	83



# Chapter 1

## Introduction

### 1.1 ARUM project

*ARUM (Adaptive production management) is a collaborative project within the European Commission (EC) "Factory of the Future" initiative and is funded under the 7th Framework Program.[3]*

The project main goal is to develop an Information Communication Technology (ICT) solution to manage complex small lot and ramp-up productions. The management of ramp-up productions is a complex task that requires monitoring and analysis of large amounts of data and their interrelationships.

Ramp-up production could be described as the phase between the product prototype release and the smooth production of the final product line. Ramp-up is the period, where a new product is being manufactured and it is often characterized by process experimentations, improvements and design changes. The production in ramp-up has often a lower output rate than the targeted "standard" production due to numerous unexpected disturbances (defective material design, inefficient processes, etc.). This rate is increased as the tested production processes are proven or modified accordingly[1].

Two companies were selected according to top level requirements and objectives set by the project: Airbus (ramp-up in aircraft industry - use case #1), and Iacobucci (small lot production, use case #2). The project outcome should be an ICT product that supports end users (decision makers) in production planning, monitoring and management.

ARUM focuses on establishing strategies to manage unexpected events in ramp-up and the mitigation of their negative impact on the running production.

The following top level objectives were defined for the project [3]:

- ARUM focuses on production ramp-ups and small lot productions to reduce costs and lead times.

- ARUM improves production systems by means of increased flexibility, autonomy, robustness and energy efficiency.
- ARUM delivers novel business strategies.
- ARUM delivers novel ICT systems for integrated control and dynamic optimization with scalable architecture.
- ARUM delivers novel tools for automation control.
- End-to-end integration the ARUM solution with legacy systems and information aggregation across existing legacy systems.
- ARUM demonstrates operational and economic benefits against today's process automation and control solutions.
- ARUM uses multiagent system approach.
- ARUM offers means for production planning and control in a predictive and real-time mode.

## 1.2 Objective of the thesis

The aim of the thesis was to develop the Scenario designer - a managerial tool within the ARUM project (initially, the tool focuses only on use case #1 - Airbus), that supports monitoring, configuration, visualization and scheduling of aircraft production in ramp-up.

The tool focuses on the following functionality:

- Arrangement of pre-production environment (assembly line layout definition)
- Assembly line configuration (setup of assembly stations - human and non-human resources assignment, station cycle time setup, etc.)
- current production monitoring
- integration with Factory network designer(FND)[6]

### 1.2.1 Former task

The original task was to analyse (requirements refinement), design and implement a graphical user interface that would fulfil the pre-requirements from early project phases, focusing on use case #1 - Airbus aircraft ramp-up production. However, after the requirements refinement, the project revealed a significant amount of additional functionality that had to be taken into account. Therefore, the decision of implementing only high fidelity prototypes (a sub-set of the functionality) for this thesis was made.

The Scenario designer(SD), in its final stages, will be integrated with other ARUM tools developed in parallel within the project. Nowadays, many of the tools are not yet finished and cannot be connected to or used by the Scenario designer. One of the tools to be used (its sub-functionality) by the SD is the *Factory network designer*[6], which is being developed by my colleague Ondřej Harcuba, within his master's degree thesis.

### 1.2.2 Latest work

At the date of this thesis submission, the requirements definition, the architecture and interface design of the Scenario designer and the Factory network designer (together referred as FNSD) have been developed into such stage, that allowed us to start with the implementation phase. After the requirements refinement, we were able to develop mockup prototypes of the graphical user interfaces(FNSD).

A sub-set of the FNSD design (so far as a mockup prototype) has been tested with end users at Airbus facilities(March and April 2014). A significant amount of valuable feedback was retrieved from both usability tests, which will be taken into account during the following implementation phases.

As previously mentioned, low fidelity and high fidelity prototypes (a sub-set of the overall functionality) could have been implemented so far. Nevertheless, the ARUM project naturally continues and after the submission of this thesis, the development of the Scenario designer will be finished.

## 1.3 Document outline

The first chapter contains a brief introduction into the ARUM project(1.1) and provides an overview of the main task of this thesis(1.2).

The second chapter provides a deeper view into the problem domain(ramp-up production, 2.2) and the solution that ARUM proposes(2.2.3). It also describes the sources of information(2.1) from which requirements for the SD tool were gathered. Finally, the chapter contains the functional requirements(2.3) and the functional specification(2.4) of the Scenario designer.

The third chapter contains the system architecture design(3.2), the graphical interface design (mock-ups (3.3)), a brief analysis of the technology used for the implementation(3.4) and a short sample of the high fidelity prototypes (3.5).

A brief description of the usability testing held in Airbus is presented in the fourth chapter(??) along with its output results.

The final chapter contains the conclusion of the thesis(5). It describes in brief the goals fulfilment.





# Chapter 2

## Analysis

### 2.1 Sources of information

The requirements analysis for this thesis was based on documents from previous phases of the ARUM project and from information provided by my thesis supervisor - Ing. Martin Klíma, Ph.D., who has been providing technical leadership to the project from the beginning.

An important source of information regarding the project is the documentation delivered by project partners. The document *Description of work*[10] contains a full list of documents that will be delivered along the project life cycle. From this list, two of the delivered documents were particularly important for the development of the Scenario designer:

- D2.1.1: Use-case Definition (use-cases #1 & #2)[3]
- D2.2.1: Use-case detailing and KPI setting [8]

Another source of information were multiple discussions with different project partners. ARUM is an international project in which several European companies and universities get involved. Therefore, some discussions were held remotely - through teleconferences. Some of the discussions also took place in personal meetings - visits from project partners.

The most valued source of information in later graphical interface design stages was the feedback received from end users. Graphical interface usability testing with real end users was held in Airbus facilities in two occasions - March and April of 2014. We were able to obtain a significant amount of valuable feedback and new information from these meetings/usability tests.

### 2.2 Domain description

Unlike large quantity serial production (cars, computers), the aircraft manufacturing is highly customized and the delivery of one product type (with the same configuration) often consists of only batches of 3 to 5 pieces. Small-lot productions (like air crafts or ships) invest much

more money than large quantity manufacturers in the design and the product ramp-up, considering the amount of products that are eventually sold. This is the reason why small lot manufacturers opt for ICT tools that support production management and monitoring. The main purpose is to speed up the learning phase and to reduce the time gap between the ramp-up and the standard production[1].

The developed tools should be able to handle/solve the most crucial issues that appear in the ramp-up/learning phase. Among the most important issues are:

- last-minute engineering changes
- immature high technology
- unexpected/disruptive events (e.g. nonconformities [A.1])
- changing requests from customers
- delayed delivery of parts, etc ...

### 2.2.1 Ramp-up

Ramp-up production in the aircraft industry *is characterized by a combination of a new product with a new production technology, both developed in parallel. This results in a high level of uncertainty at planning, which needs an appropriate level of mutability in the entire industrial system to react on.*[5].

The main market leaders in aircraft production - Boeing and Airbus often experience problems with delayed or even failed product ramp-ups. As they produce complex and highly customized products, they face many risks during the ramp-up. These risks are the source of a significant reduction of the planned production rate, which leads to a significant increase in the production costs. Due to the growing competition and the shorter amount of time that the manufacturers are able to invest into innovation, the number of ramp-ups is growing[1].

The ARUM solution should provide tools that will smooth the transition from single to series production in order to reach high production rates.

### 2.2.2 Needs in production management

The solution that is being developed by ARUM is primarily targeted to the decision makers in ramp-up productions. The developed system should solve the main issues that the production management have to face. Among the most important problems are:

- slow response to disruptive events
- unsupported response to changes of resources availability (changes in resources allocation is currently done "manually" without ICT tools support)

- missing or unsatisfactory evaluation of the running production
- unsatisfactory task/job scheduling (tasks scheduling should be improved in terms of calculation time and optimization of the scheduling results)
- unsatisfactory production planning (ARUM should provide tools to simulate the production in order to support decision making)

### 2.2.3 ARUM solution

The solution that ARUM offers, is to develop an intelligent Enterprise Service-Based platform(i-ESB), which will integrate a service-oriented architecture with a knowledge-based multi-agent system.

The ESB platform will collect and process information from multiple sources, such as sensors and management systems in order to provide the production managers an improved insight into the running ramp-up production. Information provided to the decision makers(production management) will be enriched by results of calculations performed by the system in order to evaluate the running production. Time, cost and risk values are some examples of the key performance indicators (KPIs) that will be analysed in order to provide the end users an objective production performance evaluation.

One of the tools that will be integrated within the i-ESB is the Scenario designer. Its main purpose will be the creation and modification of scenes(see section 2.4.1scene definition) and the monitoring of running production.

The "scene" concept was introduced in order to describe the production state and its configuration. The scenes (configured by the SD) will be used as pre-production settings that will be forwarded to the computational core, in order to obtain scheduled tasks in time(the scheduling result). The scenes will also be used to plan future production, or to play so called "what-if games". The what-if games are simulations based on the "live scene" (the scene reflecting the current production state) which simulate the occurrence of unexpected changes in the production (e.g. missing resources, nonconformities, etc.).

The Scenario designer will be part of a top-level application(FNSD) which will be used by different production executive roles (ARUM user roles). The FNSD will be divided into multiple modes. Every mode has been designed to fulfil different requirements and targets different user roles. The FNSD is described with more detail in section 3.

## 2.3 Refinement of requirements

This section contains a list of the functional requirements and the mapping between ARUM user roles and the functionality provided by the Scenario designer.

The functional requirements were taken from pre-requirements defined in early project phases. A more detailed analysis has been performed and the pre-requirements were extended, specified in more detail and formalized.

### 2.3.1 Specification of user roles

Several user roles were identified in AIB and IHF use cases[3] and formalized as ARUM user roles. The mapping between AIB and IHF roles and ARUM roles is described in the ARUM document - Deliverable 4.1.1.[9]. The following table provides mapping between ARUM user roles and the Scenario designer functionality.

ARUM role	Functionality
<b>Production/Planning Manager</b>	<ol style="list-style-type: none"> <li>1. Create a schedule for the workshop.</li> <li>2. Receive and display up-to-date information on the current production state.</li> <li>3. Setup conditions for calculation of what-if scenarios.</li> <li>4. Call ARUM computational core.</li> <li>5. View and analyse results obtained from ARUM computational core.</li> </ol>
<b>Team Leader</b>	<ol style="list-style-type: none"> <li>1. Monitor task execution.</li> <li>2. Assign tasks (WOs, jobs) to workers.</li> </ol>
<b>Station Manager</b>	<ol style="list-style-type: none"> <li>1. Create a schedule for the production line (one or multiple stations).</li> <li>2. Receive and display up-to-date information on the current production state (production monitoring).</li> <li>3. Adjust the schedule according to incoming events (create travelling work, modify WOs execution, add resources, etc.).</li> <li>4. Setup and modify the scene properties (add/remove stations, modify resources, etc. See SCENE LIFE-CYCLE for scene definition).</li> <li>5. Call ARUM computational core.</li> </ol>

Table 2.1: User roles mapping

### 2.3.2 Functional requirements

This section contains a formal analysis of the functional requirements for the Scenario designer mode and the production mode.

**2.3.2.1 Functional requirement 1.1****ID:** FR01**TITLE:** Log-in to the top-level application**MODE:** All**DESC:** All users allowed to use the SD are able to log-in via a login form after the application starts. The system should automatically load the user information and set the permissions according to the current user role.**RAT:** In order for a user to use the SD.**DEP:** None**2.3.2.2 Functional requirement 1.2****ID:** FR02**TITLE:** Load/create scene**MODE:** Scenario designer mode**DESC:** The user is able to load, duplicate or create a new scene (see section 2.4.1scene definition). The scene can be loaded from local storage or from the core ontology. The user is able to display the scene properties in the Scenario designer.**RAT:** In order to create a new scene, to modify an existing scene, and to generate a schedule based on the given conditions (the scene setup).**DEP:** FR01 [2.3.2.1]**2.3.2.3 Functional requirement 1.3****ID:** FR03**TITLE:** Edit scene**MODE:** Scenario designer mode**DESC:** The user is able to edit a scene in the Scenario designer. The user is able to modify all the scene properties:

1. Modify the number of stations in the scene
2. Modify the station properties (assigned resources, assembly plans)
3. Modify the production line flow
4. Setup the scene default time interval (cycle time)

**RAT:** In order to modify the scene properties and to generate a schedule based on the given conditions (scene setup). In order to play what if games.**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2]

#### 2.3.2.4 Functional requirement 1.4

**ID:** FR04

**TITLE:** Setup events

**MODE:** Scenario designer mode

**DESC:** The user is able to add/modify/remove events in a scene (MR, I, NC, D, etc.). This can be done by adjusting the scene schedule interactively (if exists) or through an event setup graphical component.

**RAT:** In order to generate a schedule based on the given conditions – a scene with a defined set of events. In order to play what if games.

**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2]

#### 2.3.2.5 Functional requirement 1.5

**ID:** FR05

**TITLE:** Save scene

**MODE:** Scenario designer mode

**DESC:** The user is able to store a scene that has been previously created or modified. The scene can be saved/stored locally or in the core ontology.

**RAT:** In order to store a previously created or modified scene. In case the scene is stored locally, only the creator of the scene is able to open/load it. In case the scene is stored in the core ontology, all users that are allowed to read from the core ontology are able to open/load the scene.

**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2]

#### 2.3.2.6 Functional requirement 1.6

**ID:** FR06

**TITLE:** Generate a schedule

**MODE:** Scenario designer mode, Production mode

**DESC:** The user is able to call the ARUM computational core (scheduler) with given conditions (scene setup). The user is able to define the scheduling time range (schedule for a shift, day, week, etc.) of the results. The user is able to define KPI priorities and the strategy for solution.

**RAT:** In order to obtain the scheduling results – multiple schedules. The best result/schedule (the one selected by the user) can be applied to the current production or can be stored for later use.

**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2]

**2.3.2.7 Functional requirement 1.7****ID:** FR07**TITLE:** Display scheduling results**MODE:** Scenario designer mode, Production mode**DESC:** The user is able view the scheduling results obtained from the scheduler.**RAT:** In order to analyze the scheduling results. In order to select the best result (which can be applied to the running production, or stored for later use).**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2], FR06 [2.3.2.6]**2.3.2.8 Functional requirement 1.8****ID:** FR08**TITLE:** Compare multiple scheduling results**MODE:** Scenario designer mode, Production mode**DESC:** The user is able to compare multiple results by their KPI values or by comparing the schedules graphically. The user is able to display the critical path in the schedule (Gantt chart). The user is also able to modify again the scene properties (former input) and to recall the scheduler for new results. The best result/schedule (selected by the user) can be applied to the current production or stored for later use.**RAT:** In order to allow the user to compare and analyze multiple scheduling results. In order to select the best result based on the KPIs comparison and the analysis of the Gantt chart.**DEP:** FR01 [2.3.2.1], FR02 [2.3.2.2], FR05 [2.3.2.5]**2.3.2.9 Functional requirement 1.9****ID:** FR09**TITLE:** Visualization of running production**MODE:** Production mode**DESC:** The Production/Planning or Station Managers should be able to view up-to-date information on the current production progress (current state, performance). This includes:

- A customizable dashboard start screen with the most crucial and relevant information. The displayed information might vary for each user role:
  - Status of Work orders (WOs)
  - Live schedule (Gantt chart)
  - Jobs to be executed the current day
  - Availability of resources
  - List of events (NCs, missing resources, start of jobs, etc.)
  - Shifts information

- Present management staff
- Production performance indicators (KPIs)
- Different views of the current production progress
  - Progress of Work visualized per station (perspective of a station manager)
  - Progress of Work visualized per product (perspective of a production/planning manager)
  - List of events per product
  - Gantt charts of all the production line (across all stations)
  - Human resources availability and their workload
  - Other resources availability and their workload (e.g. tools)
- Production performance information
  - Evaluation of critical path in Gantt chart
  - Evaluation of critical station (station with the longest lead time)
  - Evaluation of critical resources
  - KPIs for each station, product, and the entire production
  - Minimum of tardiness (per shift, station, entire line) without additional costs
  - Minimum of tardiness (per shift, station, entire line) with minimal additional costs
  - Minimum of tardiness (per shift, station, entire line)

**RAT:** In order to allow the user to monitor and objectively evaluate the current production progress and performance.

**DEP:** FR01 [2.3.2.1]



## 2.4 Functional specification

The following section captures the scenario designer processes. Rather than describing which requirements will be covered by the Scenario designer mode (see functional requirements 2.3), this section describes **how** these requirements will be fulfilled.

This section extends information provided by ARUM project document:deliverable D4.1.1 [9].

### 2.4.1 Scene life cycle

The Scene concept has been introduced in order to gather and describe entities taking part in the production system. The scene is an abstract entity consisting of:

- **Local scenes**, which consists of:
  - **Station(s)** – a self-contained workspace equipped with workers, tools and other resources. Products are assembled in stations.
  - **Product** - fuselage in AIB use case, coffee maker or other product in IHF use case. Products are shifted from station to station - every product has a set of work orders to be performed in the given station.
  - **Work orders** – list of work orders and jobs to be done in the local scene (station and product).
- **Production events** - changes in the production state are represented by production events. Some examples of production events follow:
  - Human resource(HR) related (missing worker, worker availability change, etc.)
  - Missing resource (MR) related
  - Nonconformity(NC) - a nonconformity is in general any critical deviation from the design/process specification.
  - Other events...
- **Station flow setup** - definition of how the products will be shifted from station to station
- **Initial human resource pool**
  - Human resources can have preferred station assignment
  - Each human resource has defined availability

The diagram in figure 2.1 shows the communication flow during the scene life cycle.

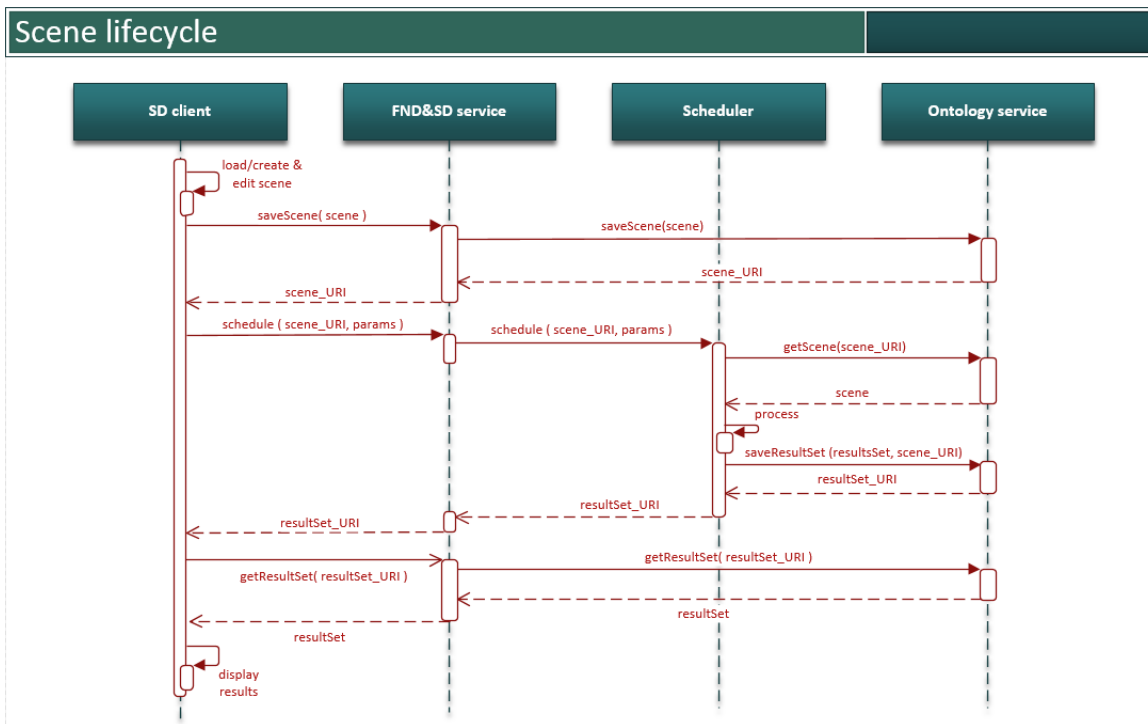


Figure 2.1: Scene life cycle sequence diagram

Scene life cycle consists of the following steps:

- **Load/create scene**
  - load existing scene
  - duplicate scene
  - new scene
- **Edit scene**
  - add/delete station
  - setup station
  - modify station flow
  - edit scene properties
- **Schedule** – set scheduling properties and call the scheduler
- **Result analysis** – scheduling results analysis and comparison
- **Apply schedules to live scene** – the user applies the best scheduling results to the live scene (current production)

The diagram in figure 2.2 shows the main scenario designer process flow. A more detailed description of each sub process is provided later in this document.

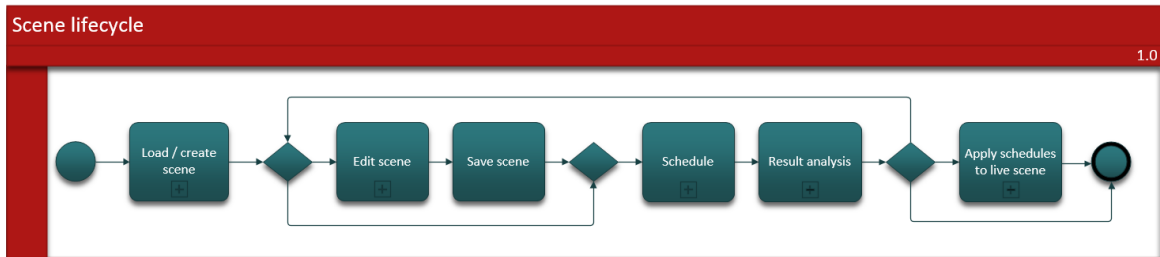


Figure 2.2: Scene lifecycle process

### 2.4.2 Load/create scene

Scene load/create (figure 2.3) consists of the following steps:

- **Create new scene** – creates a new scene from scratch
- **Load scene(s)** – loads an existing scene, or multiple scenes
- **Duplicate scene** – makes a copy of an existing scene

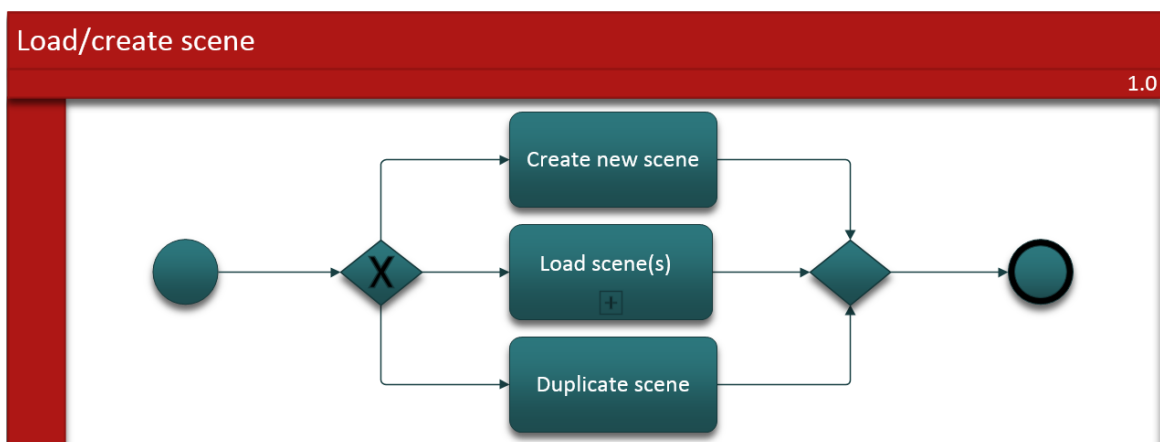


Figure 2.3: Load/create scene process

### 2.4.3 Load scene(s)

Load scene(s)(figure 2.4) consists of the following steps:

- **List existing scenes** – provides an overview of all the currently existing scenes to the user
- **Select scene(s)** – the user can select one or multiple scenes

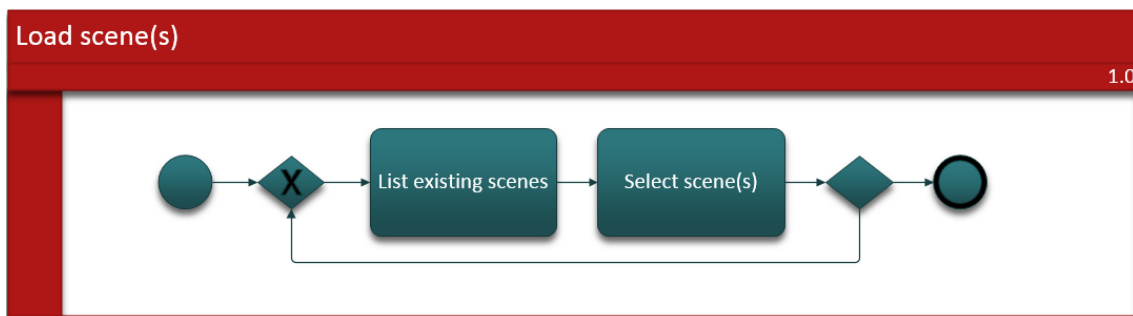


Figure 2.4: Load scene(s) process

### 2.4.4 Edit scene

Edit scene (figure 2.5) consists of the following steps:

- **Add station** – load an existing station or create a new station from scratch
- **Delete station** – remove a station from the scene
- **Edit scene properties** – edit the scene resource pool, time interval
- **Set up station** – set up the station properties ( required human resources, assembly plans )
- **Set up flow** – definition of how the products will pass through stations

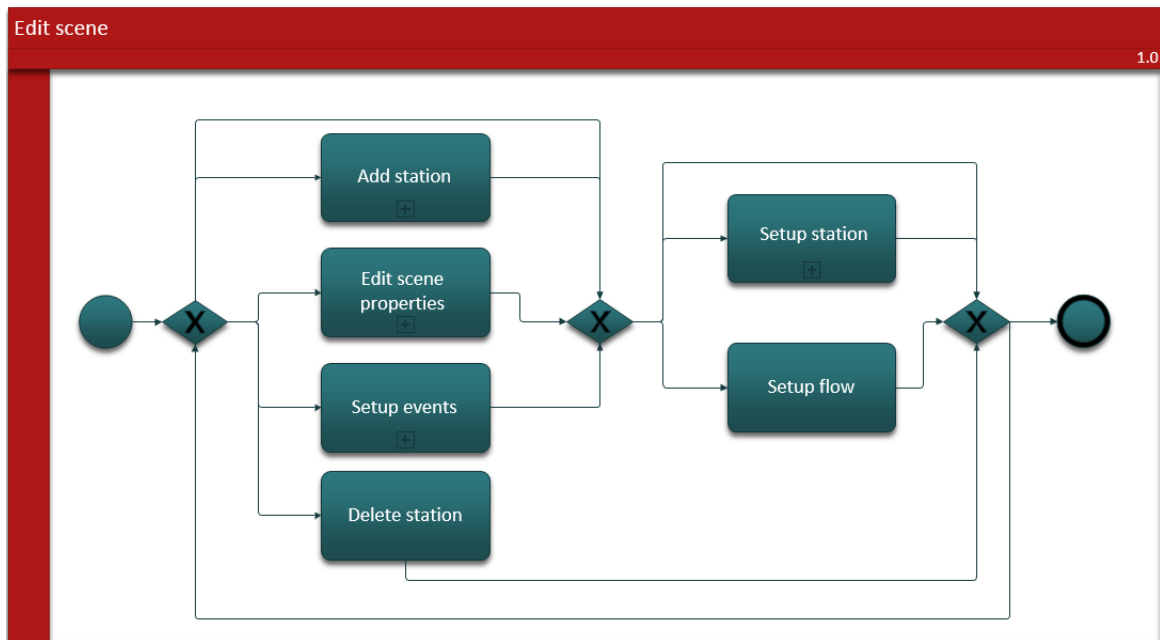


Figure 2.5: Edit scene sub-process

### 2.4.5 Add station

Add station (figure 2.6) consists of the following steps:

- **Copy from loaded scenes** – “copy paste” to duplicate a previously added station in any of the opened scenes
- **Create new station** – creates a new station from scratch

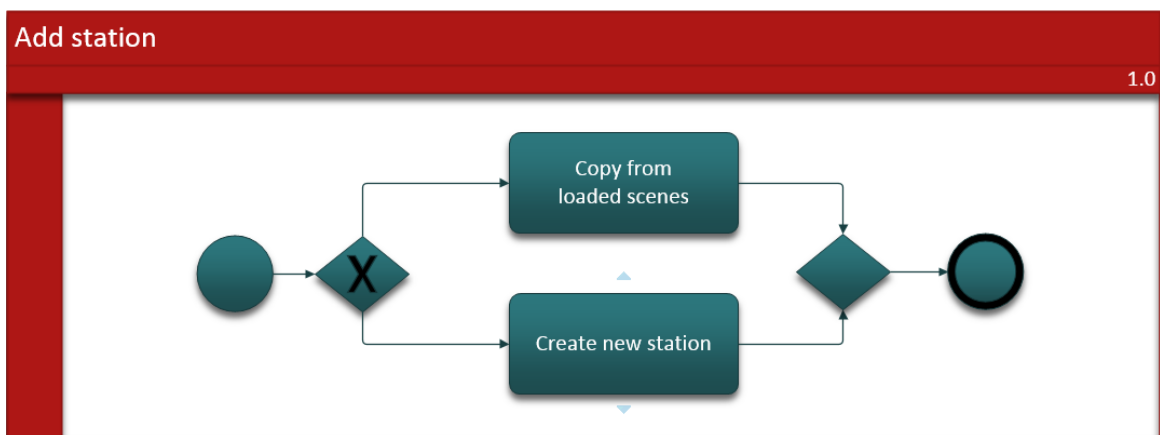


Figure 2.6: Add station sub-process

### 2.4.6 Edit scene properties

Edit scene properties (figure 2.7) consists of the following steps:

- **Setup time interval** – set the scene time interval
- **Setup initial human resource pool** – set the “default” human resource pool for the whole scene (later will be modified according to real human resource availability)

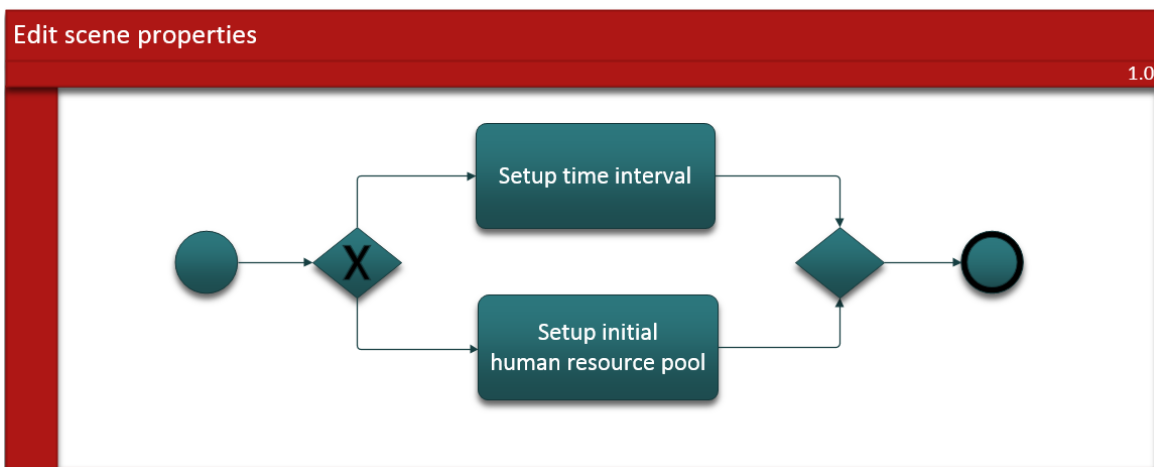


Figure 2.7: Edit scene properties sub-process

### 2.4.7 Setup events

Setup events (figure 2.8) consists of the following steps:

- **Add event** – add a new event and set its properties
- **Edit event** – edit an existing event
- **Remove event(s)** – remove one or multiple events

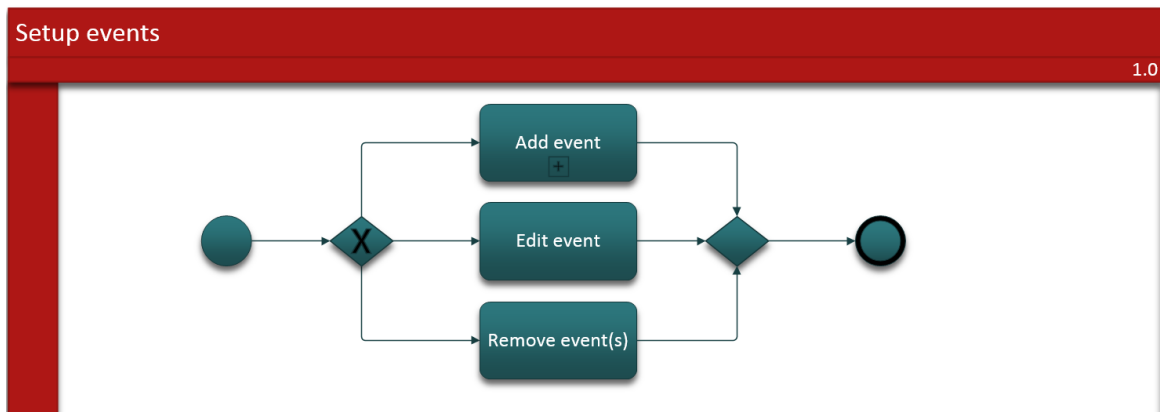


Figure 2.8: Setup events sub-process

### 2.4.8 Add event

Add event (figure 2.9) consists of the following steps:

- **New event** – insert a new event
- **Select event type** – the user selects an event type, or event category
- **Set additional parameters** – set the required parameters (event source, duration time, etc.)

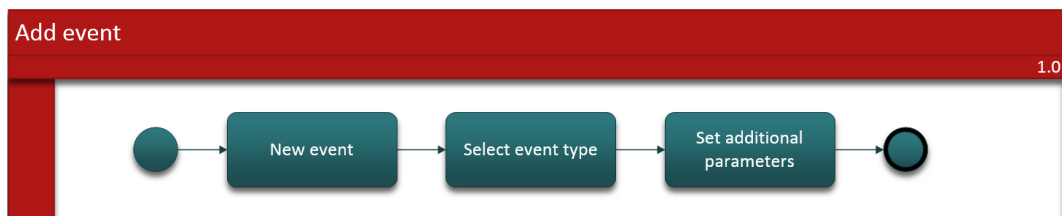


Figure 2.9: Add event sub-process

### 2.4.9 Set up station

Set up station (figure 2.10) consists of the following steps:

- **Set properties** – properties setup (planned cycle time, resources, etc.)
- **Set required human resources** – set the required set of skills
- **Set resources** – includes the setup of tools and other resources
- **Assign operations** – assign the assembly plans to the corresponding stations
- **Assign product** – assign a product to the station, or edit the currently assigned

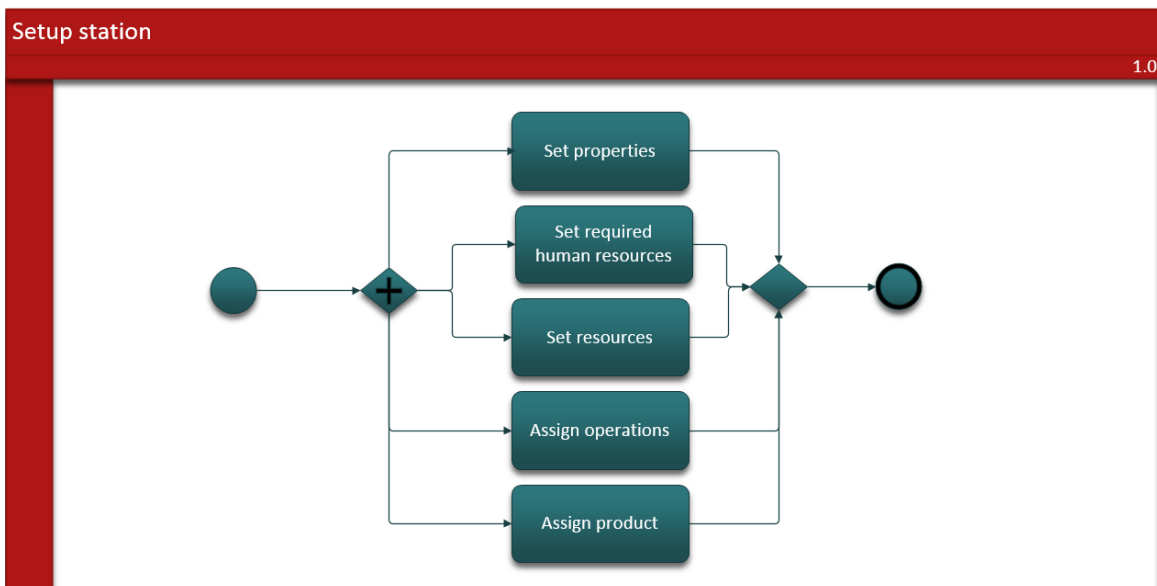


Figure 2.10: Set up station sub-process



### 2.4.10 Schedule

Schedule (figure 2.11) consists of the following steps:

- **Set scheduling properties** – sets the scheduling mode (e.g. automatic event insertion) and other properties
- **Call scheduler** – the scheduler is called in order to obtain the scheduling results

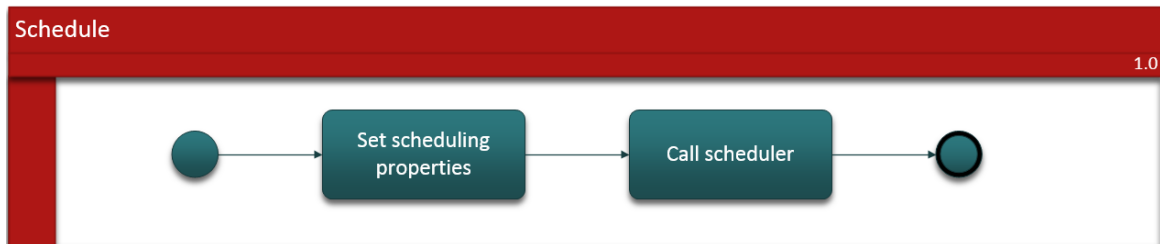


Figure 2.11: Schedule sub-process

### 2.4.11 Result analysis

Result analysis (figure 2.12) consists of the following steps:

- **Compare results by KPI** – the user can compare the schedules by their KPIs
- **Select results** – according to the KPI comparisons, the user selects one/multiple schedules for the graphical comparison
- **Schedules graphical comparison** – the user can graphically compare multiple results (display one or two Gantt diagrams)
- **Select best result** – the user selects one best result which is then applied to jobs in the current scene (assignment of start and end times to jobs)

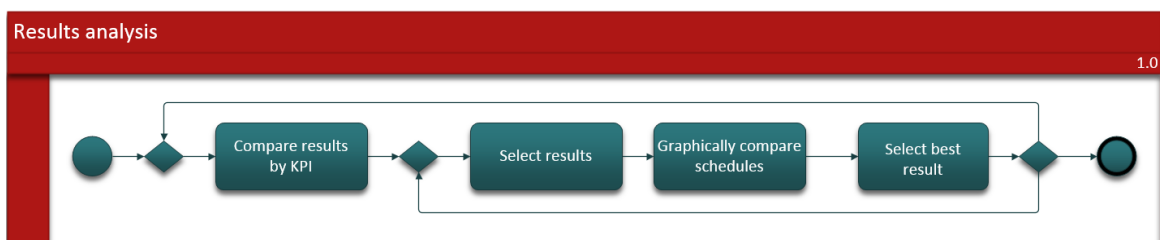


Figure 2.12: Result analysis sub process



## Chapter 3

# Design and Implementation

The following sections contains the SD graphical interface design (so far as mock-up low and high fidelity prototypes) and its description. The chapter also describes in more detail the context in which the tool is being developed (SD as a part of the FNSD top-level application, the communication flow between the tool and other ARUM components, etc.). At the end of this chapter, the analysis of technologies used for the implementation and the partial implementation of the Scenario designer itself is described.

### 3.1 Application description

At the beginning of the requirements definition (October 2013), the Factory network designer (FND) and the Scenario designer(SD) were supposed to be developed as separate applications. The FND tool development was assigned to my colleague Ondrej Hrcuba within his diploma work[6], and the development of the SD was assigned to me as my bachelor's thesis. Both ARUM components\graphical interfaces were targeted to different user roles and for different purposes:

- Factory network designer
  - **functionality:** assembly plans management (display, create, modify assembly plans[A.1])
  - **target user roles:** process managers and production/planning managers
- Scenario designer
  - **functionality:** pre-production environment configuration (assembly line composition, ramp up production simulations)
  - **target user roles:** production/planning managers and station managers

However, the fact that both applications would be used by production managers and that some functionality would be shared by both tools (production monitoring), lead to the decision of connecting these two tools through a top-level application - the FNSD.

The FNSD will contain multiple modes, which will cover the requirements for both, the SD and the FND:

- **Scenario mode:** SD functionality (scenes and simulations management)
- **Design mode:** FND functionality (assembly plans management)
- **Production mode:** FND and SD functionality (running production monitoring)

Switching between these modes will be possible through the FNSD ribbon (mode tab 3.3.1.5), which will be shared by components "inside" the FNSD. Each ARUM user will be allowed to use modes according to his permissions. All the tools/components inside the top-level application will also share the same basic layout and will follow similar interface design principles.

The decision of sheltering both tools through a common interface gives us multiple advantages:

1. The application will be easily scalable - other modes can be added without the need of changing the existing tools.
2. The FND and the SD tools are kept logically divided, but at the same time the user (if his permissions allows it) is able to switch between modes and use both functionality.
3. Permissions can be easily set - every ARUM user role will have a defined set of application modes which will be able to use (for some user roles, only application mode sub-functionality will be available).
4. All FNSD modes will share the same layout and design principles - this will make the learning phase easier for end users using multiple application modes.

Although both tools are united as the FNSD, the assigned tasks remained separated and the main topic of this thesis is the Scenario designer development.

## 3.2 Architecture of the FNSD

The following section contains the design of the FNSD system architecture. The section contains a description of components/entities being developed within the ARUM project and describes the interaction between them (data flow in time). In the following chapters, the SD client (Scenario designer) represents any managerial tool (any other managerial tool within the FNSD could be depicted instead – FND, planner, etc.).

### 3.2.1 ARUM system architecture diagram

Below is shown a general overview of the ARUM system architecture. Only relevant entities (with respect to the SD communication with other services) and communication canals are illustrated.

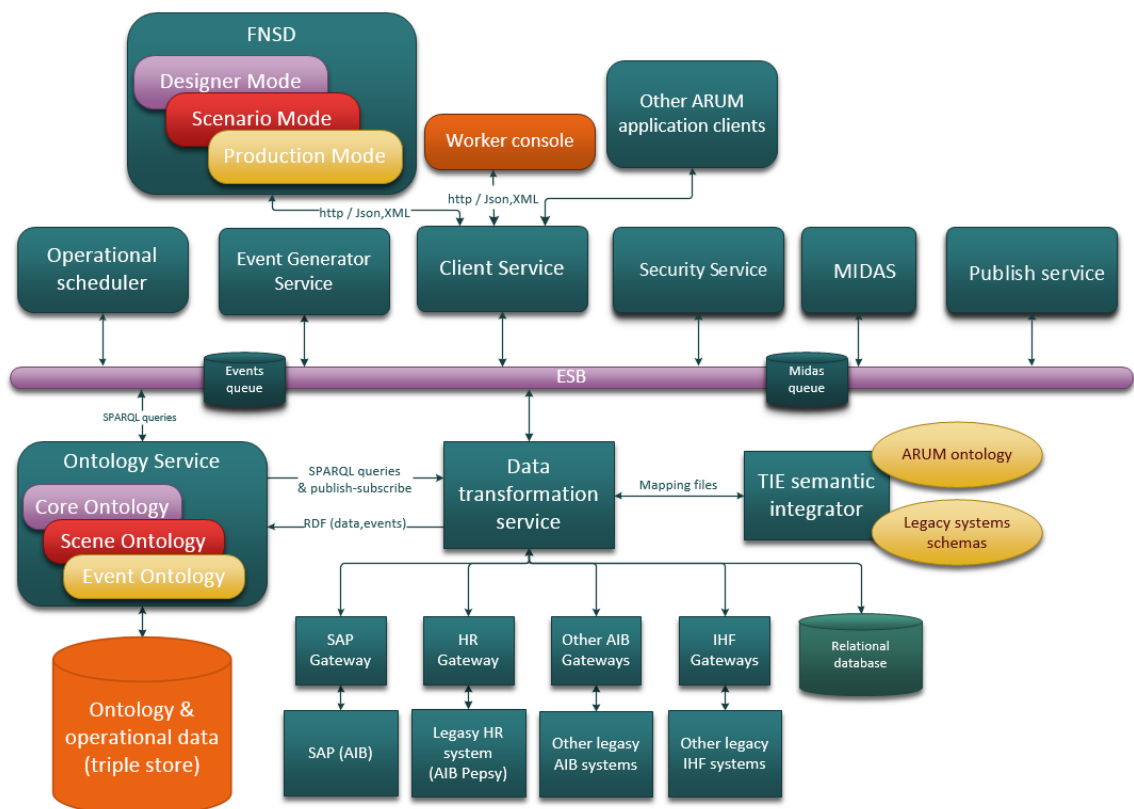


Figure 3.1: ARUM system architecture

### 3.2.2 System architecture components

- **FNSD** – represents the top-level application that contains multiple modes/tools (Design mode, Scenario mode, Production mode). It is a thin client using the Client Service for handling user requests.
- **Other ARUM application clients** – represents any other ARUM user console that will communicate within the ESB (worker console, warehouse console, etc.)
- **Client Service** – a service for the purpose of handling requests coming in and out from the FNSD or any other ARUM console. The data interchange between this component and other ARUM entities is provided via the Enterprise Service Bus (ESB).
- **Security service** – a service handling the client authentication.
- **Operational scheduler** – the ARUM computational core. A component for task scheduling/planning.
- **Event generator service** – a service used by the FNSD tool for the purpose of obtaining generated hypothetical production events on request (e.g. to perform simulations).
- **Ontology Service** – a service providing data in the “ARUM ontology format”. The data will be load-ed from the triple store or retrieved from the „Data transformation service”.
- **Data transformation service** – a service providing the transformation of data from legacy systems into the RDF format.
- **TIE semantic integrator** – a component providing mapping of files. These files are provided to the “Data transformation service” in order to transform data from legacy systems to the RDF format.
- **MIDAS** – a component with the purpose of providing analysis of disruptive production events. The component will provide data to facilitate the troubleshoot process. Specifically, in the case of the FNSD, the component will be able to provide the estimated troubleshoot time for given production event.
- **Publish service** – a service handling upcoming messages from ARUM components. After a mes-sage is received, it is forwarded it to the corresponding message “queue”. Every queue has a unique identifier.
- **Production events queue** – A component of the “publish-subscribe” messaging model. It repre-sents a queue – like data structure receiving production event mes-sages. In this case, the source of a production event can be an ARUM console, or legacy systems.
- **Midas queue** – A component of the “publish-subscribe” messaging model. It repre-sents a queue – like data structure receiving messages from Midas.
- **Relational database** – In case of AIB, the relational database is a temporary solution to store data (temporarily replaces data from legacy systems).

### 3.2.3 Communication within the ESB

There are two message delivery modes available when communicating within the ESB:

- **synchronous**
- **asynchronous**

When using the synchronous messaging, the service invoker waits for the response as opposed to the asynchronous messaging where the response (if sent) is delivered as a new message to the action processing pipeline.

The responses (and fault messages) can be routed to other services, different from the original sender using the **ReplyTo** (and **FaultTo**) message headers. If there is a specified (other than the sender) **ReplyTo** header while using synchronous messaging, the service invoker will throw a **responseTimeoutException** since the response message will be routed elsewhere and no response will be sent to the original sender. The same applies to the **FaultTo** header in case of errors[4].

If a service sends a message (regardless whether the message is a request or a response), it should always include its Logical EPR (end point reference – addresses an entity communicating within the ESB) in the **From** header.

Because the recommended interaction pattern within ESB is based on one-way message exchange (asynchronous communication), responses to messages are not necessarily automatic: it is application dependent as to whether or not a sender expects a response. As such, a reply address (EPR) is an optional part of the header routing information and applications should be setting this value if necessary. However, in the case where a response is required and the reply EPR (ReplyTo EPR) has not been set, the ESB supports default values for each type of transport[4].

#### 3.2.3.1 Errors handling

It is possible that an error occurs during the transmission/reception or processing of messages sent within the ESB. If an error occurs, the ESB will route any faults to the EPR mentioned in the **FaultTo** field of the incoming message. If this is not set, then it will use the **ReplyTo** field or, failing that, the **From** field[4].

Sequence diagrams in the following chapters depict how ARUM components interact (communicate) with one another and in what order.

### 3.2.4 Load scene

The diagram in figure 3.2 shows the communication between the SD client and the ontology service during the “Load scene” process.

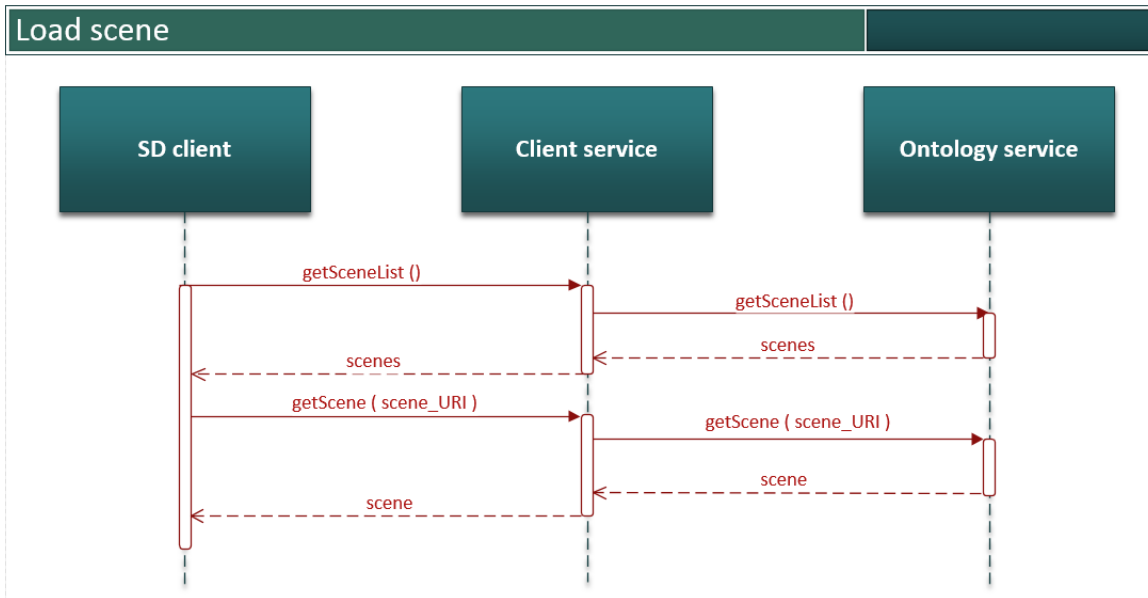


Figure 3.2: Load scene

Method	Parameters	Returns	Sync async
getSceneList()	n\ a	A list of URIs referencing all existing scenes (topmost scenes).	sync
getScene( scene_URI)	<b>scene_URI</b> – URI address of the required scene	A scene object defined by the given scene_URI (topmost scene) with basic properties: timeInterval, attribute, localScenes. Every localScene has product, station and cycleTime properties.	sync

Table 3.1: Load scene data flow methods



### 3.2.5 Create scene

The diagram in figure 3.3 shows the communication between the SD client and the ontology service during the “create scene” process.

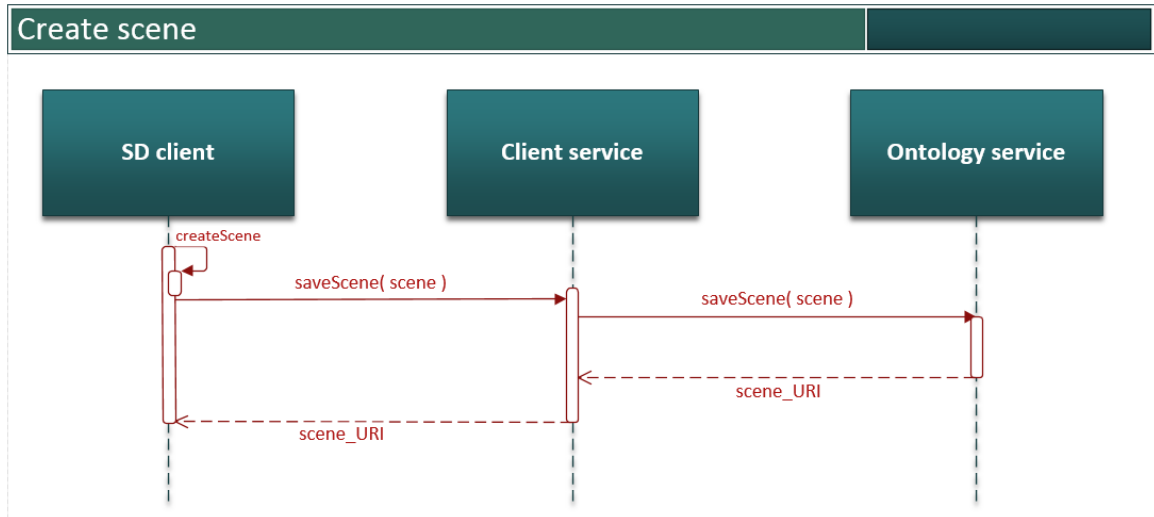


Figure 3.3: Create scene

Method	Parameters	Returns	Sync async
saveScene( scene )	<b>scene</b> – a scene object (top most scene) in the RDF format. The scene object contains local scenes (every local scene has a station with cycle time and a product), list of resource pools and time interval.	URI of the stored scene if the scene was successfully stored, failure message with the reason (String) otherwise.	sync

Table 3.2: Create scene data flow methods

### 3.2.6 Update scene

The diagram in figure 3.4 shows the communication between the SD client and the ontology service during the “update scene” process.

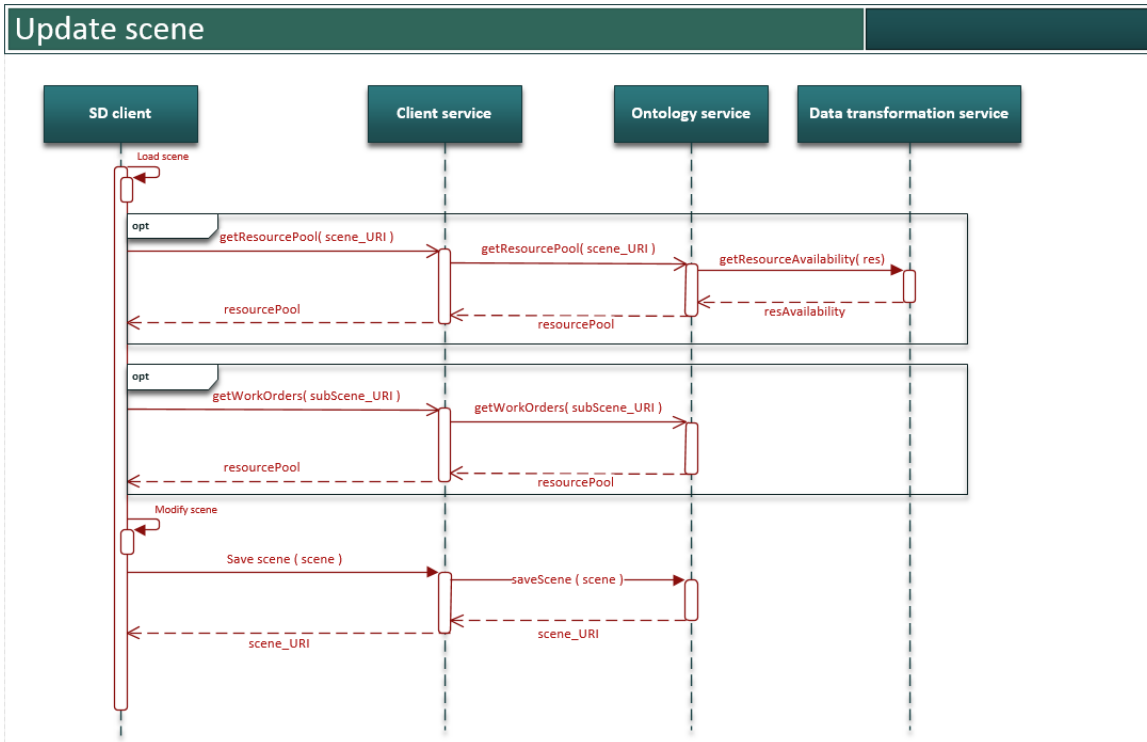


Figure 3.4: Update scene

Method	Parameters	Returns	Sync async
getResourcePool( resPool_URI )	<b>resPool_URI</b> – URI of the required resource pool	Resource pool (human and non-human) object with all the corresponding resource properties (availability, HR-skills).	sync
getWorkOrders( subscene_URI )	<b>subscene_URI</b> – URI of the subscene linked to the required work orders.	A list of all work orders linked to the given sub-scene URI.	sync

saveScene( scene )	<b>scene</b> – a scene object in the RDF format. The scene object contains a local scene definition (station with cycle time and product), list of resource pools and time interval.	URI of the stored scene if the scene was successfully stored, failure message with the reason (String) otherwise.	sync
--------------------	--	---	------

Table 3.3: Delete scene data flow methods

### 3.2.7 Delete scene

The diagram in figure 3.5 shows the communication between the SD client and the ontology service during the “delete scene” process.

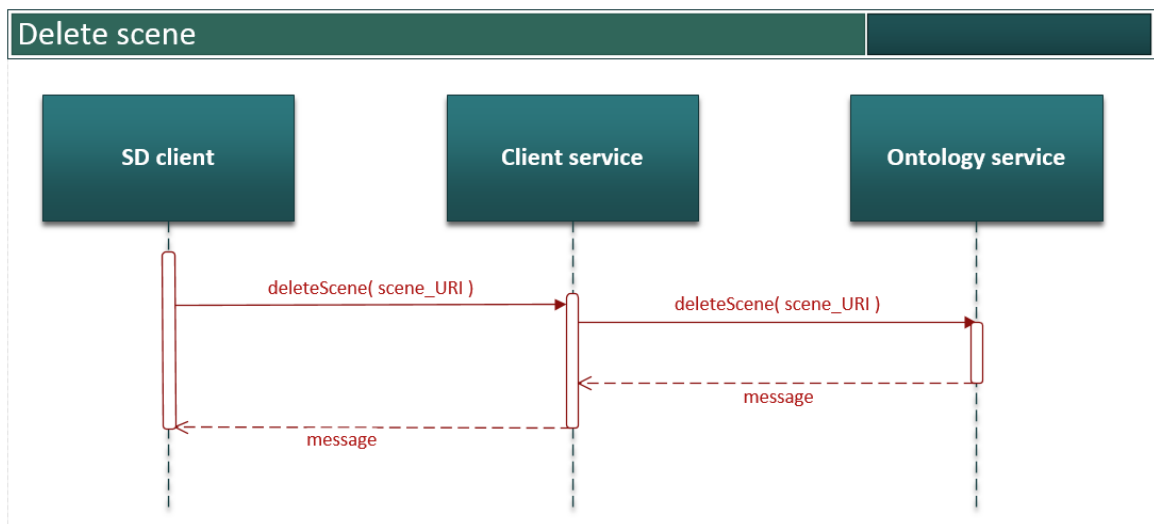


Figure 3.5: Delete scene

Method	Parameters	Returns	Sync async
deleteScene( scene_URI )	scene_URI – URI of the scene to be deleted from the core ontology	Confirmation message in case of successful scene removal, error message with the reason (String) otherwise. The ontology service deletes the scene and its properties (cycle time, time interval, and re-source pools references) and the sub-scenes and their properties (hasProduct, hasStation, hasWorkOrder).	sync

Table 3.4: Update scene data flow methods

### 3.2.8 Scheduling

The diagram in figure 3.6 shows the communication between the SD client and the ontology service during the “scheduling” process.

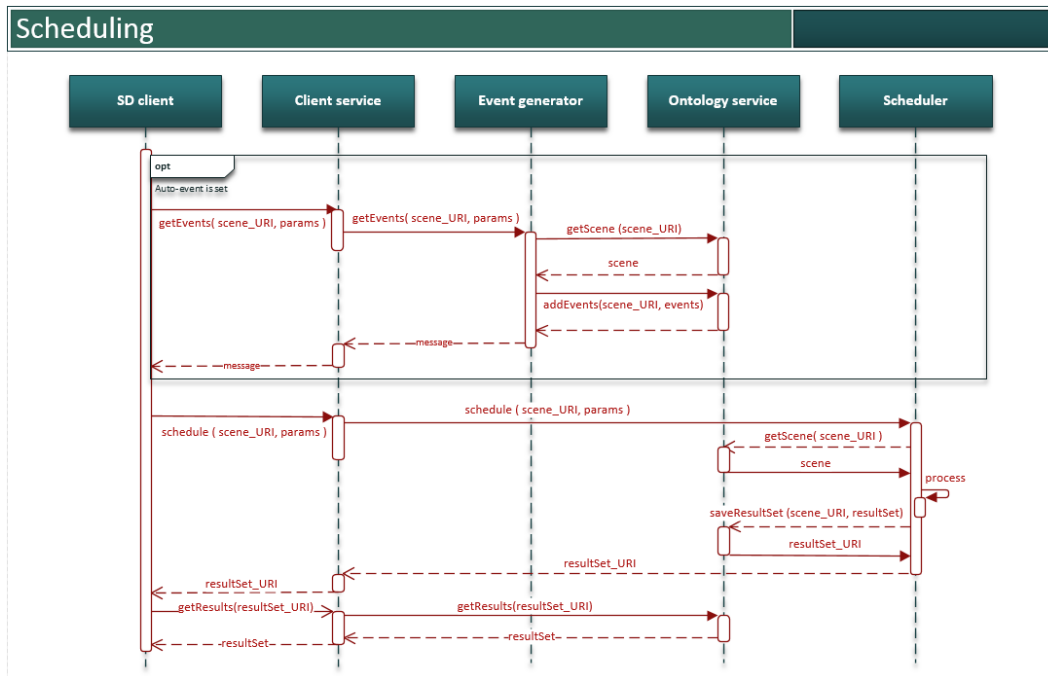


Figure 3.6: Scheduling

Method	Parameters	Returns	Sync async
getEvents( scene_URI, params )	<b>scene_URI</b> - the scene URI for which events will be generated <b>params</b> - additional parameters to specify the required events	message – confirmation message informing that events have been inserted into the scene, error message with the reason (String) otherwise	async
getScene( scene_URI)	<b>scene_URI</b> – URI address of the required scene	A scene object defined by the given scene_URI	sync
addEvents ( scene_URI, events)	<b>scene_URI</b> – URI address of the scene, for which events were generated <b>events</b> – a list of events to be stored in the core ontology (and linked to the scene defined by the given scene_URI	message – confirmation message informing that events have been inserted into the scene, error message with the reason (String) otherwise	sync
schedule ( scene_URI, params )	<b>scene_URI</b> - the scene for which schedules will be generated <b>params</b> - additional parameters required for the scheduler	<b>resultSet_URI</b> – URI of the generated result set (scheduling results)	async
saveResultSet( resultSet )	<b>resultSet</b> – the result set to be stored in the core ontology	<b>resultSet_URI</b> – URI of the generated result set (scheduling results)	sync
getResults( resultSet_URI)	<b>resultSet</b> – URI of the required result set	Result set object containing the scheduling results	sync

Table 3.5: Scheduling data flow methods

### 3.2.9 Scheduling involving MIDAS

The diagram below (3.7) shows the communication flow between the SD client, the ontology service, MIDAS and the scheduler during the rescheduling. In this particular case, MIDAS provides the approximate troubleshoot time of a given disruptive event.

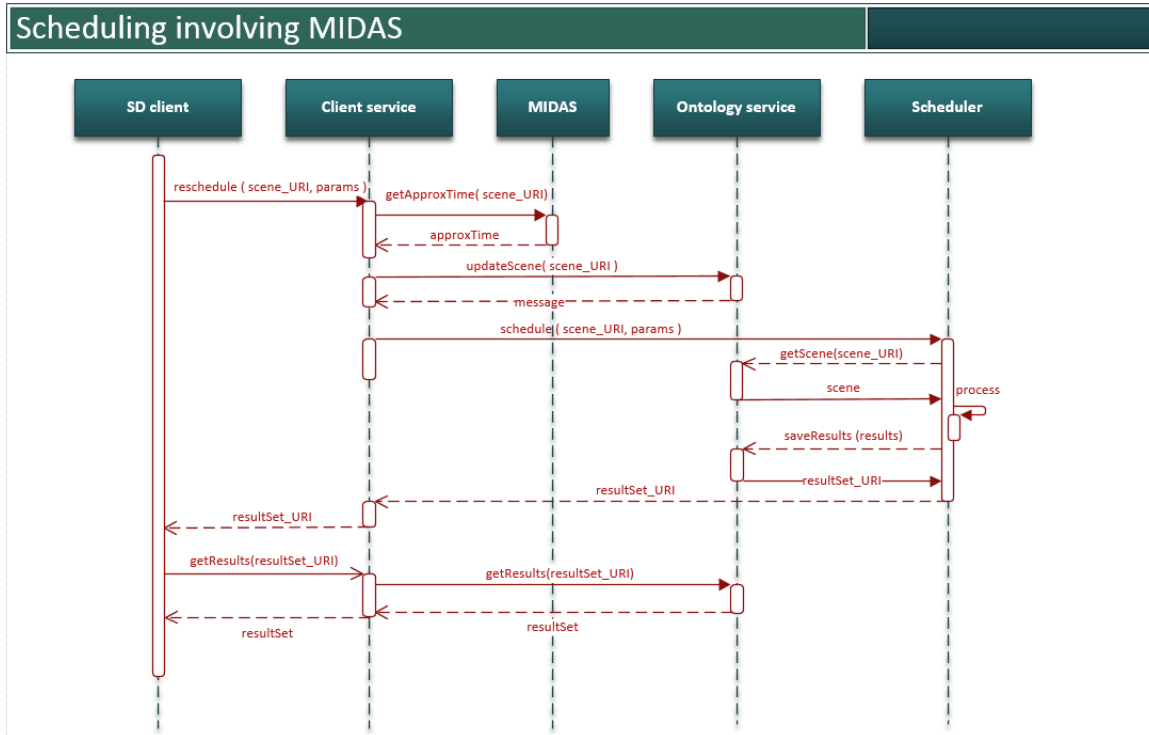


Figure 3.7: Scheduling involving MIDAS

Method	Parameters	Returns	Sync async
reschedule ( scene_URI, params )	<b>scene_URI</b> - the scene for which a new schedule will be calculated <b>params</b> - additional parameters required for the scheduler	resultSet_URI – URI of the generated result set (scheduling results)	async
getApproxTime( scene_URI )	<b>scene_URI</b> - the scene for which the approximate troubleshoot time should be calculated	approxTime – the approximate troubleshoot time for a given scene (e.g. blocked by a disruptive event)	sync

updateScene( scene_URI	<b>scene_URI</b> – the scene that will be updated (according to the approximate troubleshoot time)	message – confirmation message informing that the scene has been updated, error message with the reason (String) otherwise	sync
schedule ( scene_URI, params )	<b>scene_URI</b> - the scene for which schedules will be generated <b>params</b> - additional parameters required for the scheduler	<b>resultSet_URI</b> – URI of the generated result set (scheduling results)	async
saveResultSet( resultSet )	<b>resultSet</b> – the result set to be stored in the core ontology	<b>resultSet_URI</b> – URI of the generated result set (scheduling results)	sync
getResults( resultSet_URI)	<b>resultSet</b> – URI of the required result set	Result set object containing the scheduling results	sync

Table 3.6: Scheduling involving MIDAS data flow methods

### 3.3 Scenario designer graphical interface

The following sections contain the design and description of the Scenario designer(SD) graphical user interface (GUI) design.

From the beginning of the development, effort was made to follow processes of a user-centered design. The SD and the FND GUI (FNSD) were first developed as mock-up prototypes, in order to facilitate the adjustments required by end users. The FNSD mock-up prototypes have been finished and the most crucial functionality of the production mode has been tested in two occasions with end users at Airbus facilities. Feedback obtained from both usability tests will be applied in further implementation phases.

The following sections contain mock-up versions (low fidelity) of the SD GUI design. The mock-ups were developed using the prototyping software "Balsamiq mockups".

Common (FNSD) components and layout are described in section 3.3.1. The Scenario designer mode components are described in section 3.3.2. A brief description of the production mode is provided in section 3.3.3.

#### 3.3.1 FNSD components and layout

This section describes components that are used in all application modes within the FNSD. Common components have the same functionality and behaviour, but contain different information/data depending on the application context(current mode).

##### 3.3.1.1 Login window

The login window (figure 3.8) is the first view displayed to the user after starting the application. It consists of a panel with login and password input fields, Login button, Exit button and the company or application logo. By pressing the Login button the application verifies the user's login and password (user authentication). Depending on whether the application can authenticate the user, two options are possible:

- If the authentication is successful, the application starts and the main window is displayed.
- If the verification is not successful, the application will display an error message under the input fields and the user will be asked to re-entry his login credentials.



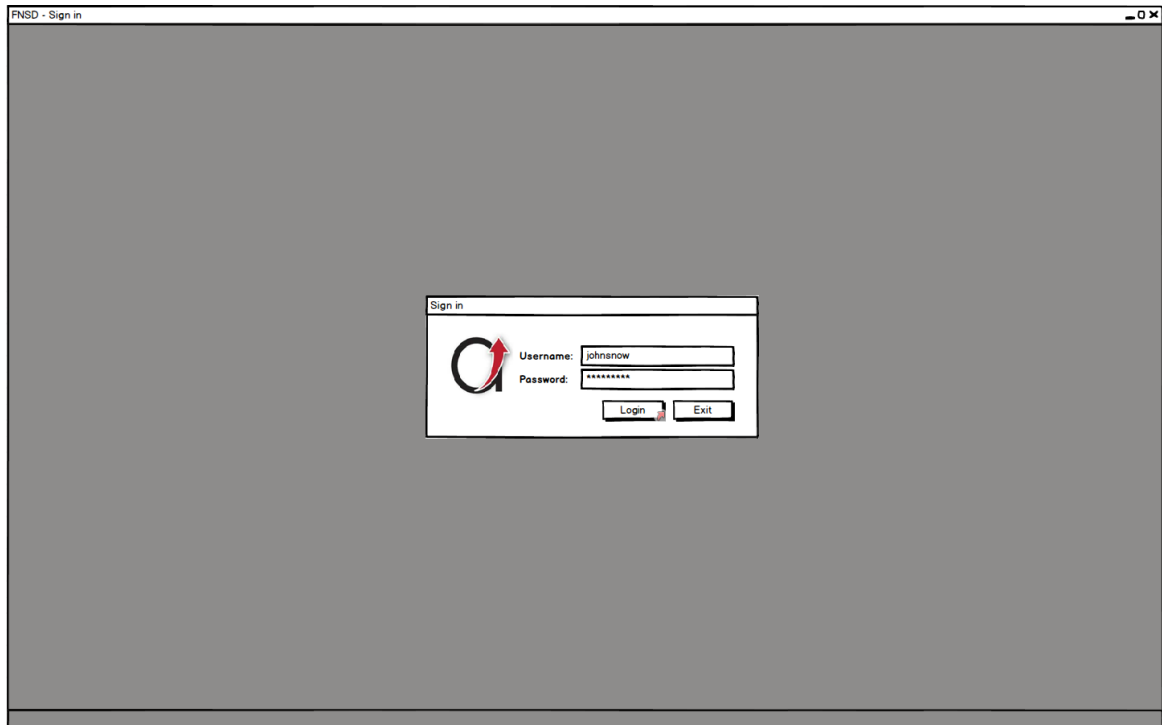


Figure 3.8: FNSD login window

### 3.3.1.2 Main window

After the user successfully logs in, the system starts in the application mode that corresponds to the user (his ARUM user role [9]). The application also loads the users last saved preferences (components layout, data displayed in components, etc.). Different modes and initial views correspond to different ARUM user roles. The following table provides mapping between modes, user roles and views:

ARUM user role	Application mode	Initial view
Station manager	Production mode	Dashboard[6]
Process manager	Design mode	Assembly plan designer[6]
Production\planning manager	Scenario designer mode (see section 3.3.2)	Scene management (see 3.3.2.1)

Table 3.7: User roles to application modes mapping

Regardless the starting mode and view, the main window (figure 3.9) is always displayed as the uppermost component and it can also contain other components\panels. Some components in the main window are common to all application modes and some are mode-specific.

Components that are common to all modes are described in the following sections: Quick access tool bar (3.3.1.4), Ribbon (3.3.1.5), Main content panel (3.3.1.6), Properties panel (3.3.1.8), Navigation panel (3.3.1.9), Search Component (3.3.1.10), Tables (3.3.1.11).

The layout of the components inside the main window can be modified (a component can be resized, collapsed, relocated, docked, etc.) in order to adjust the view to the user preferences. Further description of the layout modification is provided in the corresponding component sections.

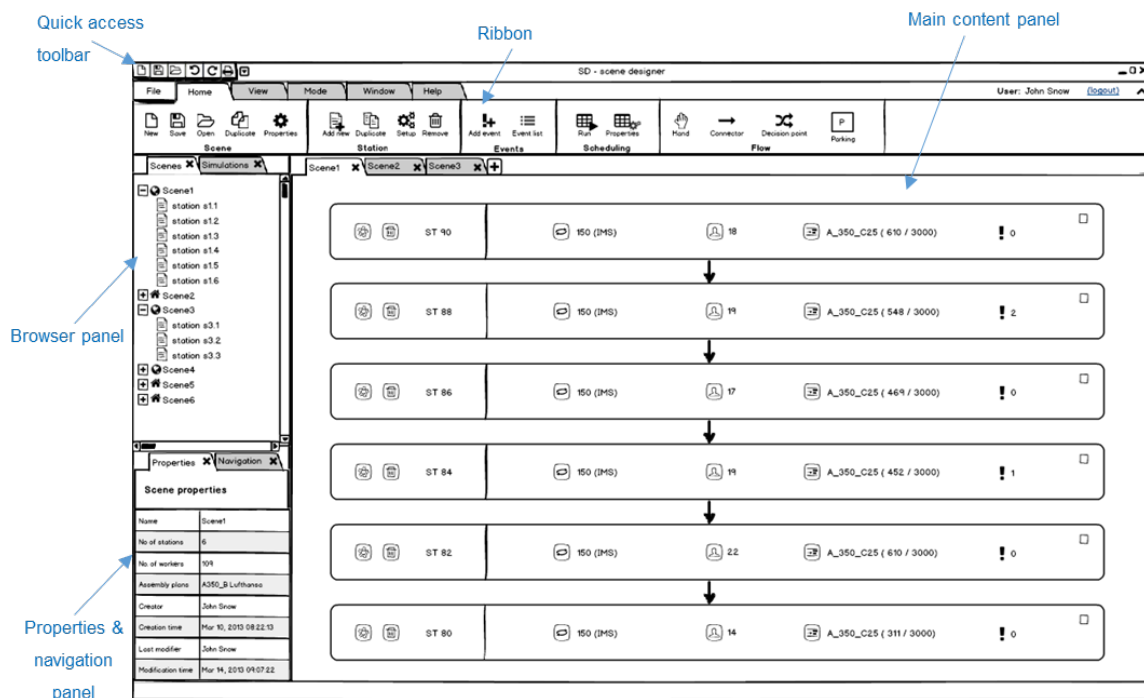


Figure 3.9: FNSD main application window

### 3.3.1.3 Sub windows

The main window can have multiple sub-windows, which are called **dialog** windows. Dialog windows are in-dependent sub-windows meant to display information or components apart from the main application window. Most dialog windows present warning, error, or notification messages but can also contain other components like setup wizards (e.g. Station setup wizard 3.3.2.9). Every dialog window has a title bar and can contain the following control buttons:

- **minimize** button – hides the dialogue window (not always supported)
- **maximize** button – the dialogue is enlarged to fit the entire screen (not always supported)

- **close** close button – closes the dialog window (always supported)

A dialog window can be modal, which means that when is visible, it blocks user input to all other windows in the application, except for any windows created with the dialog window as their owner. The modal dialog captures the window focus until it is closed, usually in response to a button press. An example of a modal dialog window is shown in figure 3.10.

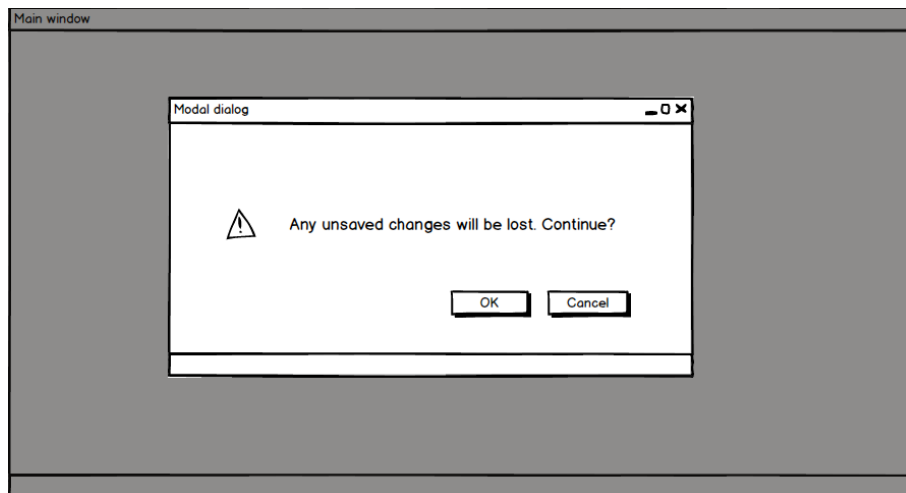


Figure 3.10: FNSD modal window

#### 3.3.1.4 Quick access tool-bar

The quick access tool-bar (figure 3.11) is located in the upper left corner of the main window (see 3.3.1.2). The quick access tool-bar can be customized in order to display different number of buttons. This can be done by clicking the tool-bar settings button. The quick access tool-bar is present in all application modes and therefore, the tool-bar buttons have different functionality depending on the current mode.

Description of the quick access tool-bar follows (buttons are described from left to right):

- **New** - creates a new assembly plan in Design mode, or a new scene in the Scenario designer mode. This button is disabled in the Production mode.
- **Save** – saves data displayed in the main content panel (3.3.1.6). The following outcomes are possible:
  - The tab with the focus in the main content panel (3.3.1.6) contains modified data, but it has been saved in the past – saves the data to its original location.
  - The tab with the focus in the main content panel contains modified data and it has not been saved before (e.g. contains a new scene) – opens a file chooser (3.3.1.12) which allows the user to select the target location and save the file.

- **Open** - Opens a file chooser (3.3.1.12) allowing the user to open a file and display it in the main content panel (3.3.1.6). Depending on the current application mode, the user can open assembly plans – design mode, scenes and simulations – scenario designer mode, etc.
- **Undo** – the application provides tracking of up to 100 changes. The undo button allows the user to undo his previous action related to data modification.
- **Redo** – allows the user to redo an action that he previously undid.
- **Print** – opens a print dialog. The dialog allows the user to print data currently displayed in the main content pane (3.3.1.6).
- **Tool-bar settings** – displays a small panel where the user can modify the tool-bar preferences – i.e. he is allowed to choose buttons that will be displayed in the quick access tool-bar.

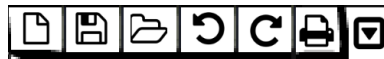


Figure 3.11: FNSD quick access tool-bar

### 3.3.1.5 Ribbon

The ribbon (figure 3.12) is designed to help the user to quickly find the control buttons required to complete a task. The buttons are organized in logical groups, which are collected together under tabs.

The ribbon content (displayed buttons) varies according to the current tab and application mode. It can also display or hide buttons according to the current application context. Disabled buttons in the ribbon indicate that their functionality is not supported in the current context. If two modes contain the same buttons (same functionality) they must have the same location in the ribbon, regardless the mode.

The ribbon is locked to the upper part of the main window(3.3.1.2) and it cannot be repositioned. The ribbon is collapsible by pressing the “hide ribbon” button (▲). When the user presses this button the ribbon is hidden and the space previously covered by it is newly populated by the components below (e.g. main content panel, browse panel, etc.).

The username of the currently signed-in user is displayed in the upper right corner of the ribbon. The username serves also as a link allowing the user to log out from the application. There are 2 possible actions after clicking on the username link:

- If the user has saved all his previous work he is safely logged out from the application and the login window is displayed (see figure 3.8).

- If some tab in the main content panel (3.3.1.6) contains unsaved data, the application will display a warning dialog asking the user to save the unsaved data.

Button sections in the ribbon are divided into the following tabs:

- **File tab** – general application options
- **Home tab** – buttons to control the most utilized tools in the current mode. The buttons in this tab are only applicable to the current mode.
- **View** – control buttons related to window/panel view modes.
- **Mode tab** – contains switching mode buttons (see figure 3.12)

The mode switch tab allows the user to switch between several application modes. Before switching the mode, the user will be asked to save all his unsaved work. After switching the mode, the interface is readjusted according to the selected mode. The application should remember the last used application mode for every user.

The user is allowed to switch to modes that he has permissions for. Description of the application modes follows:

- **Design mode** - used by process managers for creating and editing assembly plans[6].
- **Production mode** - used by production/planning managers and station managers for monitoring the current production status and progress.
- **Scenario designer mode** – used by production/planning managers for creating and editing scenes and performing production simulations.
- **Other modes** - the development of the "strategic planner" is currently running and the tool will be integrated into the FNSD in the future.

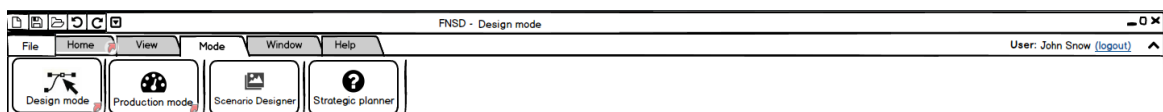


Figure 3.12: FNSD ribbon - mode switch

### 3.3.1.6 Main content panel

The main content panel (figure 3.13) is located by default in the middle of the main window and serves as the main working area. The panel can display data or other components in multiple tabs. The user is allowed to open, close or swap tabs.

Every tab has the following properties:

- is labeled with the name of the file or the component it contains
- has a close button next to the tab name that allows the user to close the tab – if the user attempts to close a tab with unsaved work, a warning dialog is displayed
- if the tab contains unsaved work/data – an asterisk is displayed next to the name of the tab
- can be active (its content is visible), inactive (all not active tabs) or disabled (is locked and cannot become visible)
- the close tab button is active only on active tabs, to prevent an accidental close of a tab while clicking a tab or switching their position

Content in the main content panel can be loaded in the following ways:

- Through the **ribbon** (3.12) – by pressing a button that opens a file or displays a component
- Through the **quick access tool-bar**(3.11):
  - By clicking on the “**new**” **button** – creates a new file (assembly plan, scene, etc.)
  - By clicking on the “**open**” **button** – displays a file chooser (3.19) dialog allowing the user to open a file in a new tab.
- Through the **new tab button** in the main content panel(figure 3.13) – opens a new tab
- Through the **browser panel** (3.14) – any file displayed in the browser panel can be opened in a new tab by double-clicking on the file.

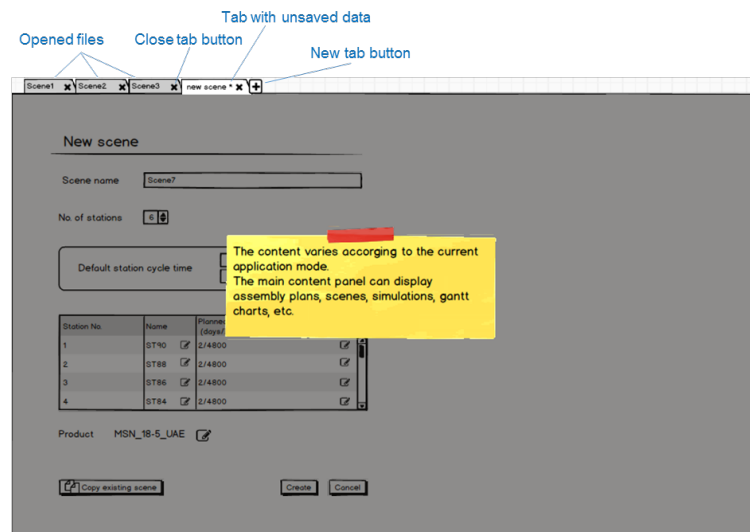


Figure 3.13: FNSD Main content panel

### 3.3.1.7 Browser panel

The browser panel is located by default on the middle left side of the main window (3.9), above navigation panel.

The browser panel component displays hierarchical data in a tree form. The panel is organized into several tabs, each containing different type of data (scenes, simulations, etc.). Different tabs are displayed in different modes (e.g. SD – scenes and simulations, FND – Assembly plans and scenes, etc.). A detailed description of the displayed data in the SD mode is provided in the corresponding section (3.3.2).

The tree inside the browsing panel displays data vertically. Each row displayed by the tree contains exactly one item of data, which is called a node. Every tree has a root node from which all nodes (children) descend.

A node can either have children or not. Nodes that can have children are called branch nodes. Nodes that cannot have children (atomic data that has no successors – e.g. a result-item in simulations tab) are called leaf nodes.

Branch nodes can have any number of children. The user can expand and collapse branch nodes by double clicking on them – making their children visible or invisible.

The browsing panel provide horizontal and vertical scroll bars, in order to display large amount of data (the width and depth of the tree can exceed the visible section of the browsing panel).

The browsing panel can also display notifications – e.g. notification for the user about a new result returned by the scheduler. After the user logs in to the application, the last view/state of the trees in the browsing panel is displayed.

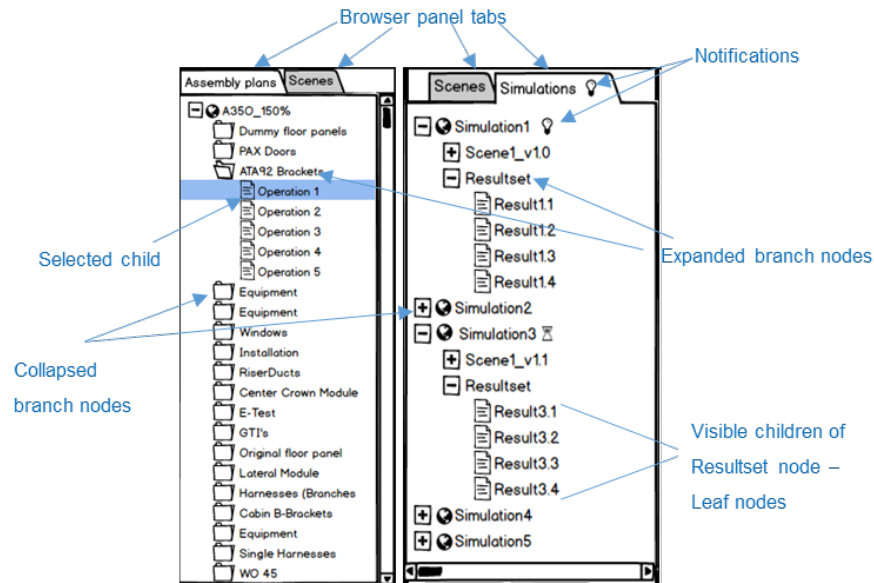


Figure 3.14: FNSD Browser panel

### 3.3.1.8 Properties panel

The properties panel (figure 3.15) is located by default on the bottom left side of the main window (3.9), below the browser panel.

Scene properties	
Name	Scene1
No of stations	6
No of workers	109
Assembly plans	MSN_16-02_UAE
Creator	John Snow
Creation time	Mar 10, 2013 08:22:13
Last modifier	John Snow
Modification time	Mar 14, 2013 09:07:22

Figure 3.15: FNSD Properties panel

The properties panel will be, by default, folded together with the navigation panel, creating a single component. The user can switch between tabs in order to view the properties panel or the navigation panel. Both, the properties and the navigation panel can be resized and relocated and they can be unfolded apart. The user is also able to close a single panel, or



both of them. The user will be able to restore their view through control buttons in the ribbon ("view" tab).

The properties panel will display detailed information about objects (assembly plans, jobs, scenes, stations, etc.) currently visible in the main content panel (3.3.1.6). In case the objects in the main content panel consists of other objects, the user will be able to display their properties in the properties panel by clicking on them (giving them focus).

### 3.3.1.9 Navigation panel

The navigation panel (figure 3.16) is located by default on the bottom left side of the main window (3.9), below the browser panel.

The navigation panel will be, by default, folded together with the properties panel (3.3.1.8) creating a single component. The user can switch between tabs in order to view the navigation or the properties panel. Both, the properties and the navigation panel can be resized and relocated and they can be unfolded apart. The user is also able to close a single panel, or both of them. The user will be able to restore their view through control buttons in the ribbon ("view" tab).

The properties panel will display (if supported) a content thumbnail of the tab currently holding the focus in the main content panel (3.3.1.6). The *visible section box* (see figure 3.16) indicates the section is currently visible in the main content panel.

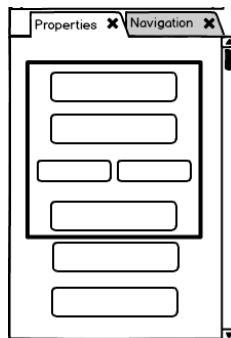


Figure 3.16: FNSD Navigation panel

The navigation panel supports the following navigation:

- **Assembly plans** – navigation through large amount of work orders by moving the visible section box.
- **Scenes** – navigation through scenes with large amount of stations by moving the visible section box.

The thumbnail displayed in the navigation panel displays a complete preview of a given object (the whole scene, i.e all stations of a scene). The user is allowed to navigate across the object by moving the visible section box – the view in the main content panel will change accordingly.

### 3.3.1.10 Search component

The search component (figure 3.17) forms part of other components containing large amounts of data – typically tables. It allows the user to find specific data inside other components.

The search component is located (if the component supports search) on the right top side of the given component. The user inserts a string into the search input field and by pressing enter, the first matching data item is high-lighted (the view is adjusted accordingly). If no items are found, an information dialog is displayed. The user is allowed to move to the next/previous matching item by clicking on the **find next**\**find previous** buttons (see figure 3.17).

Description of the search component follows:

- Search options (⚙️) – opens a small panel allowing the user to set the search preferences (e.g. case-sensitive search, whole words only, etc.).
- Find previous (◀️) – finds the previous occurrence and highlights the matching item.
- Find next (▶️) – finds the next occurrence and highlights the matching item.



Figure 3.17: FNSD Search component

### 3.3.1.11 Tables




Tables (figure 3.18) are typically used to display large amount of serial data. Data in a table can be **filtered**, **searched** and **sorted**.

In addition to data, tables can contain other components like check boxes, drop down lists, editable fields, icons, links, control buttons etc.

Tables can provide the following functionality:

- **Filtering** – data in a table can be filtered by several criteria, which are fixed and defined for every table. The filtering can be done by selecting one of the options in the filtering drop down list – e.g. figure 3.18 shows an example of the work orders assignment component[6], in which the user can filter the displayed work orders. The user is able to display unassigned work orders only, work orders to be assigned, etc.
- **Sorting** – columns in the data table can be sorted by clicking on the column sort button (☒), if supported. The column sort button is displayed on the right top corner of every column that supports column sorting.
- **Simple sorting** – columns that support column sorting can be sorted in ascending or de-scending order by clicking on the column sort button. When the user clicks on the column sort button, the column is sorted in ascending order and the buttons appearance is changed (▲). Another click sorts the column in descending order (▼).
- **Multi-column sorting** – allows the user to sort the table by multiple columns at the same time. The user can activate multi-column sorting by “shift” clicking on the column he wants to add to the sort.
- **Searching** – if a table supports data searching, the search component (3.3.1.10) is displayed on the upper right corner of the table (see figure 3.18).

ST 90 - work orders assignment

View by:     

ID	Name	Predecessors	ST 90	ST 88	ST 86	ST 84	ST 82	ST 80
5678945678	☕ Dummy floor panels	0 Q	☐	☐	☐	☐	☐	☐
2345677879	☑ PAX Doors	1 Q Installation	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲
2314356780	☑ Operation 1	1 Q Installation	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲
0086756453	☑ Operation 2	1 Q Installation	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲
3424567543	☑ Operation 3	2 Q Operation 1, Operation 2	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲
2312345678	☕ ATA 92 Brackets	1 Q Installation	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲	☐ ▲

Figure 3.18: FNSD Table example

### 3.3.1.12 File chooser

The file chooser (figure 3.19) component is displayed in a dialog (see sub-windows 3.3.1.3) and allows the user to find and choose a file from the local or remote (core ontology) storage. The file can be a scene, station, simulation, etc., depending on the current context (application mode and view).

The file chooser contains the following sections:

- The **file path bar** allows the user to select the storage path of files that will be displayed in the files table – there are two root options: remote storage (core ontology) or local storage.

- Some file choosers might provide a station **flow preview**, or other additional info on the right side of the **files table**. The file chooser example in figure 3.19 displays the station flow of selected scene.
- General additional information is provided below the **files table**. The visible information belongs to the currently selected scene (scene 1 in the mock-up).
- **Open button** – opens the selected content (scenel in the mock-up) in a new tab in the main content panel (3.3.1.6).
- **Cancel button** – closes the file chooser dialog and returns to the previously opened panel. No changes are applied.

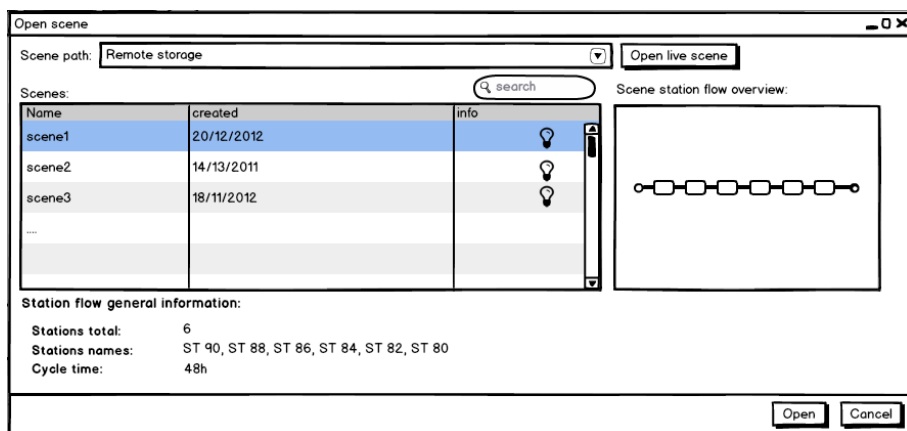


Figure 3.19: FNSD File chooser example

### 3.3.2 Scenario designer components

The following GUI mock-ups illustrates the functionality provided by the Scenario Designer. All screens are described and explained in detail below the mock-ups.

#### 3.3.2.1 Scene management

The scene management view (figure ) is the first view provided to the user after switching from other modes to the SD mode (see ribbon mode tab 3.3.1.5).

This view contains all necessary tools for managing scenes and its properties, stations and its properties (resources), events, simulations. The description of each component displayed in the scene management view follows.

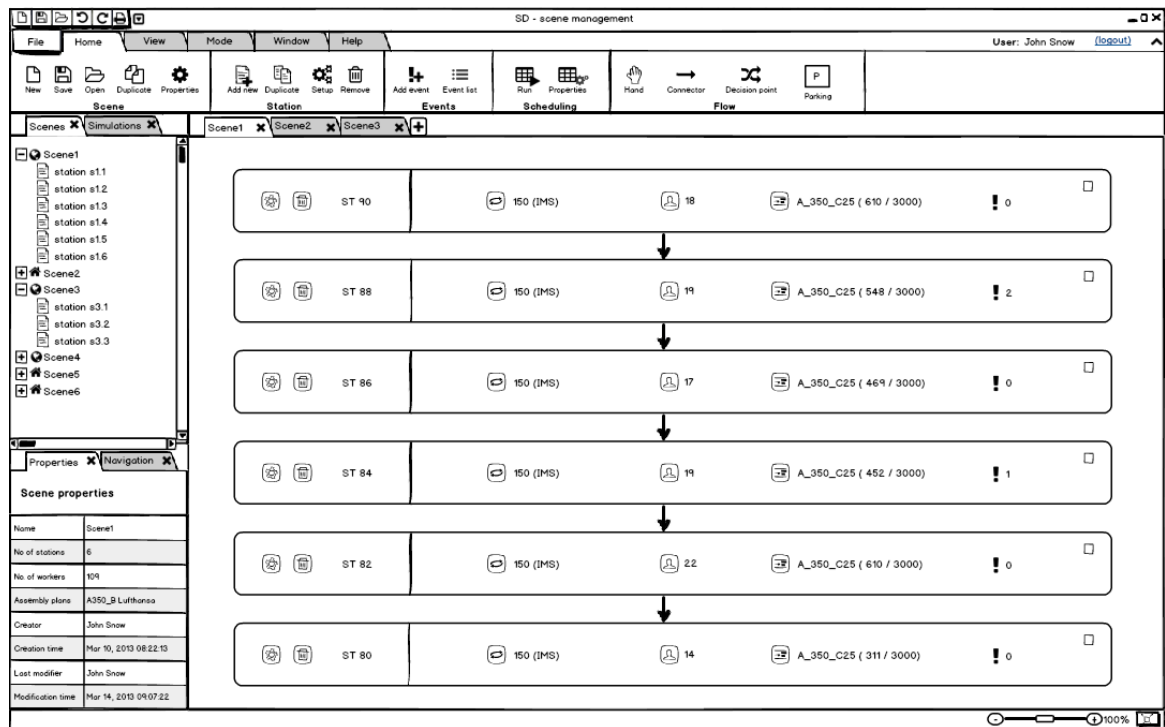


Figure 3.20: Scene management - AIB

### 3.3.2.2 Scenario designer ribbon

The ribbon general description is provided in section 3.12.

Description of the scenario designer ribbon (figure 3.21) groups follows:

- Scene group
  - New scene button (📄) - creates a new scene from scratch. The new scene is opened in the main content panel (3.3.1.6).
  - Open scene button (📁) – opens a file chooser dialogue (3.3.1.12). The file chooser allows the user to select and open a scene stored locally or in the core ontology. The scene will be opened in the main content panel.
  - Scene settings button (⚙️) – opens the scene settings dialog. The scene settings dialog allows the user to view and modify the opened scene properties.
  - Duplicate scene button (📄📄) – opens a file chooser dialog. The file chooser allows the user to select and duplicate a scene stored locally or in the core ontology. The duplicated scene will be opened in the main content panel.
  - Save scene button (💾) – opens a save file dialog. The save file dialog allows the user to save the opened scene locally or in the core ontology

- Station group

- Add new station button (📄➕) – adds a new station from scratch to the currently opened scene.
- Duplicate stations button (📄📄) - opens a file chooser dialog. The file chooser allows the user to select and duplicate a station stored locally or in the core ontology. The duplicated station will be added to the opened scene in the main content panel.
- Station settings button (⚙️) – opens the station settings/properties view.
- Delete station button (🗑️) – removes the selected stations from the opened scene.

- Events group

- Add new events button (📄➕) – adds a new event (see add events dialogue).
- View list of events button (☰) – opens a list of all events of the opened scene.

- Scheduling group

- Scheduling button (📅) – calls the scheduler, after the scene has been set up.
- Default scheduling properties button (📅⚙️) – opens the default scheduling properties win-dow - the user can set the preferred scheduling properties (automatic insertion of events, preferred scheduling strategy)

- Flow section

- Hand button (👤) – provides a tool to adjust the placement of the stations in the scene. The hand tool also allows the user to select one or multiple stations.
- Connector button (➡️) – the connector tool provides a connection between stations in order to define their flow.
- Decision point (🔄) – inserts a decision point. This element is required for the IHF case, in order to dynamically decide the product flow (e.g. defining the product flow in case of parallel stations).
- Parking point (🅐) – inserts a parking point. This element is required for the IHF case – it represents a location where products can be temporarily placed.

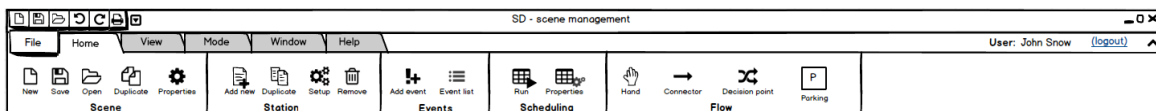


Figure 3.21: Scenario designer ribbon

### 3.3.2.3 Scenes and simulations browser

The scene/simulations browser provides an overview of the currently existing scenes and simulations. A double click on a scene/simulation opens the scene/simulation in a new tab in the main content panel 3.3.1.6.

The browser panel contains the following sections:

- Scenes tab (figure 3.22)
  - Every scene can contain one or multiple stations. The stations can be viewed in the scenes browser by expanding the scenes ("+" button left to the scene name). Every scene is marked with an icon, informing the user about their current storage location: Remote (☉) icon indicates that the scene is stored in the core ontology. Local (🏠) icon indicates that the scene is stored locally.

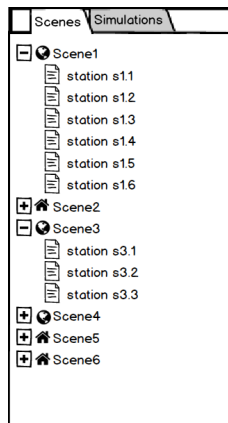


Figure 3.22: Scenario designer browser panel - scenes tab

- Scenes tab (figure 3.23)
  - Every simulation contains a scene and a result set belonging to that scene. The icons left to the simulation name indicates whether the simulation is stored in the core ontology, or locally (as in the scenes tab).
  - The “scheduling in progress” (⌚) icon next to a simulation (simulation 3) indicates, that the result set of the simulation is not yet ready to be displayed – the scheduler is still computing the results.
  - The “info icon” (🔍) in the Simulations tab indicates that a new result set has been returned by the scheduler. The result set is ready to be displayed by the user. Once the user opens and views the result set – the “info icon” disappears from the simulations tab.
  - The info icon (🔍) next to the Simulation name indicates that the result of that simulation is ready to be displayed. Once the user opens and views the result set – the “info icon” disappears (simulation1 in the mock-up).

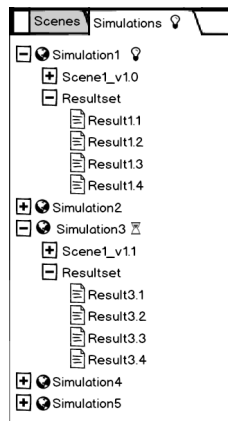


Figure 3.23: Scenario designer browser panel - simulations tab

### 3.3.2.4 Scenario designer properties and navigation panel

The properties/navigation panel (figure 3.24) provides additional information about objects being displayed in the main content panel. It is located on the lower left corner of the main window (3.9).

Scene properties	
Name	Scene1
No of stations	6
No of workers	109
Assembly plans	MSN_16-02_UAE
Creator	John Snow
Creation time	Mar 10, 2013 08:22:13
Last modifier	John Snow
Modification time	Mar 14, 2013 09:07:22

Figure 3.24: Scenario designer properties panel

The navigation tab (figure 3.25) provides an overview of the scene flow. The purpose of this tool is to provide a top view of the scene flow and facilitate the navigation through the opened scene.

The rectangle indicates the currently visible scene section. This section can be moved in any direction in order to navigate across the scene (the opened scene in the main content panel will move accordingly) Note: the Navigation tab above, displays a scene with 7 stations, 2 of which are parallel (does not correspond to the screen in section main content panel)



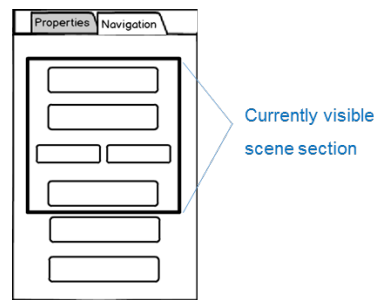


Figure 3.25: Scenario designer navigation panel

### 3.3.2.5 Station overview

The station main view displays in brief the station properties. The station view has 2 sections:

- Left section contains(from left to right):
  - Settings button – opens a setup wizard to setup the station properties (planned cycle time, human resources, tools, plans assignment)
  - Delete button – deletes the station from the scene
  - ST 90 – ID of the displayed station
- Right section contains (from left to right):
  - Cycle time label – indicates the planned cycle time (150 Industrial minutes)
  - Human resources label – indicates the number of assigned workers to the station (18 workers)
  - Assigned assembly plans label – indicates the assigned assembly plans and displays the number of work orders assigned to the station (610 work orders out of 3000)
  - Selection checkbox – allows the selection of multiple stations (multiple delete, copy).



Figure 3.26: Station overview

### 3.3.2.6 Add stations from multiple scenes

The dialog below (figure 3.27) allows the user to add station(s) from one or multiple scenes into the scene opened in the main content panel.

The user opens a new scene by clicking on the “new tab” button – this opens a dialog and the user can select and open existing scenes (stored in the core ontology, stored locally or the live scene).

Once the user has opened the scenes, he can collect the stations by clicking on the collect station button (📌) left to the station name. The collected stations will be displayed in the bottom table (Collected stations table).

The right tabbed panel contains additional scene information in order to provide a more detailed view of the opened scene:

- The flow tab preview displays the scenes station flow – here, the stations are highlighted according to the selection in the Opened scenes table. (Station 2 is highlighted both, in the opened scenes table and in the flow preview)
- The details tab displays the scene properties (scene properties tab and flow preview).

The (💡) button in the collected stations table opens a dialog displaying the assigned resources .

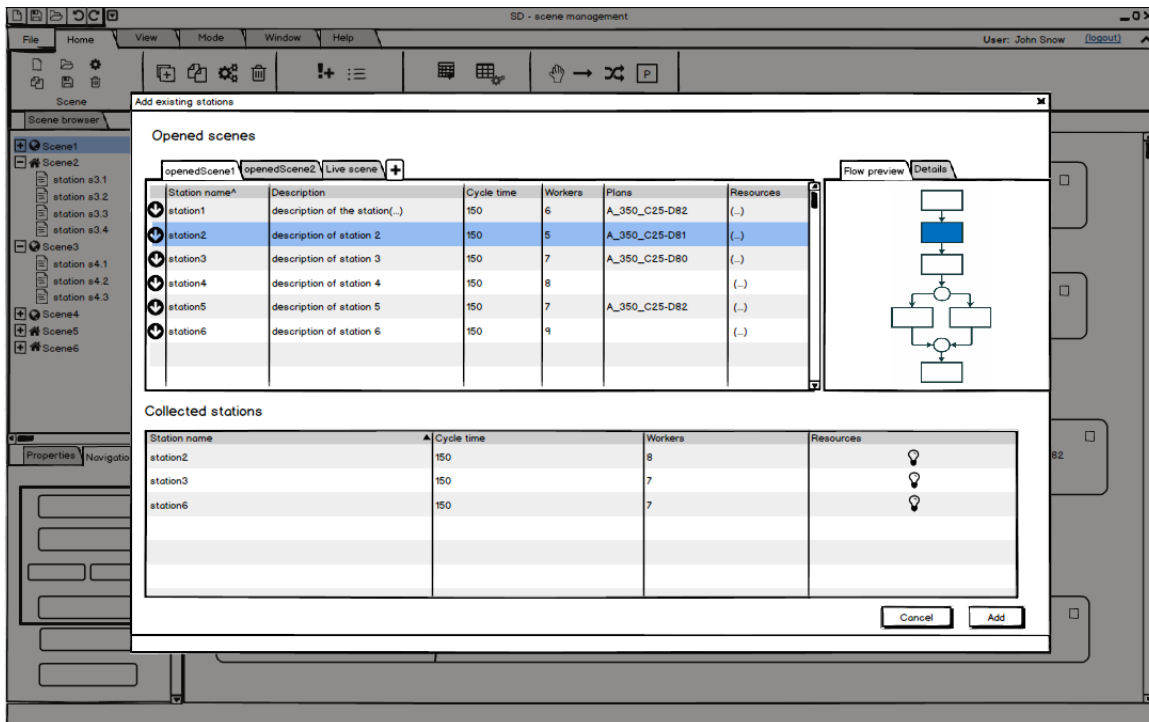


Figure 3.27: Add stations from multiple scenes

### 3.3.2.7 Add events

After pressing the add events button in the SD ribbon, the SD interface displays a dialogue (see figure 3.28) that allows the user to add events to a scene . The “add events dialog” is divided into three sections:

Left section – events catalogue: the event catalogue allows the user to browse through event categories and to add new events into the scene (the user is allowed to add multiple events – displayed in the “added events” table). The text box below the event catalogue contains text describing the selected event (Process nonconformity in the mock-up). The "add" button adds the selected event to the right table (added events).

Middle section – “added events” table. Contains all events inserted by the user.

Right section – Event setup – components inside this section are dynamically rendered, according to the selected event type in the “added events” table (the required parameters to be set up will differ depending on the event type).

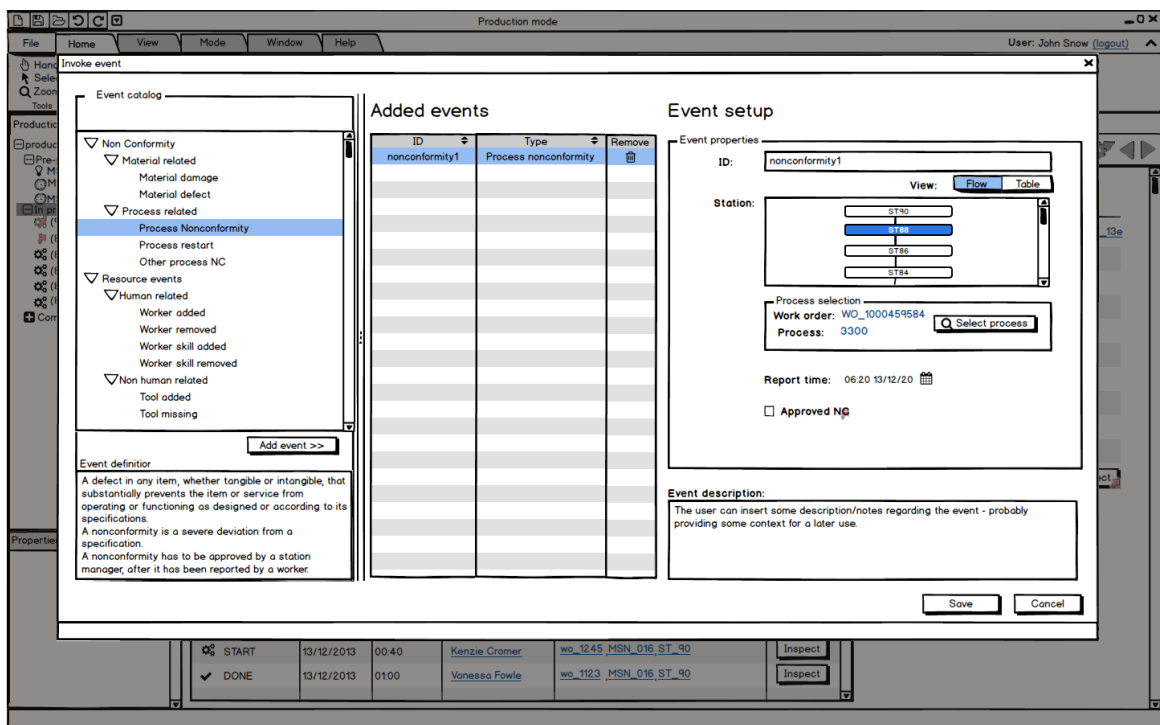


Figure 3.28: Add events

### 3.3.2.8 New scene

The “new scene” component (see figure 3.29) allows the user to setup general scene properties. It is opened in a new tab in the main content panel after pressing the “new scene” button in the scenario designer ribbon.

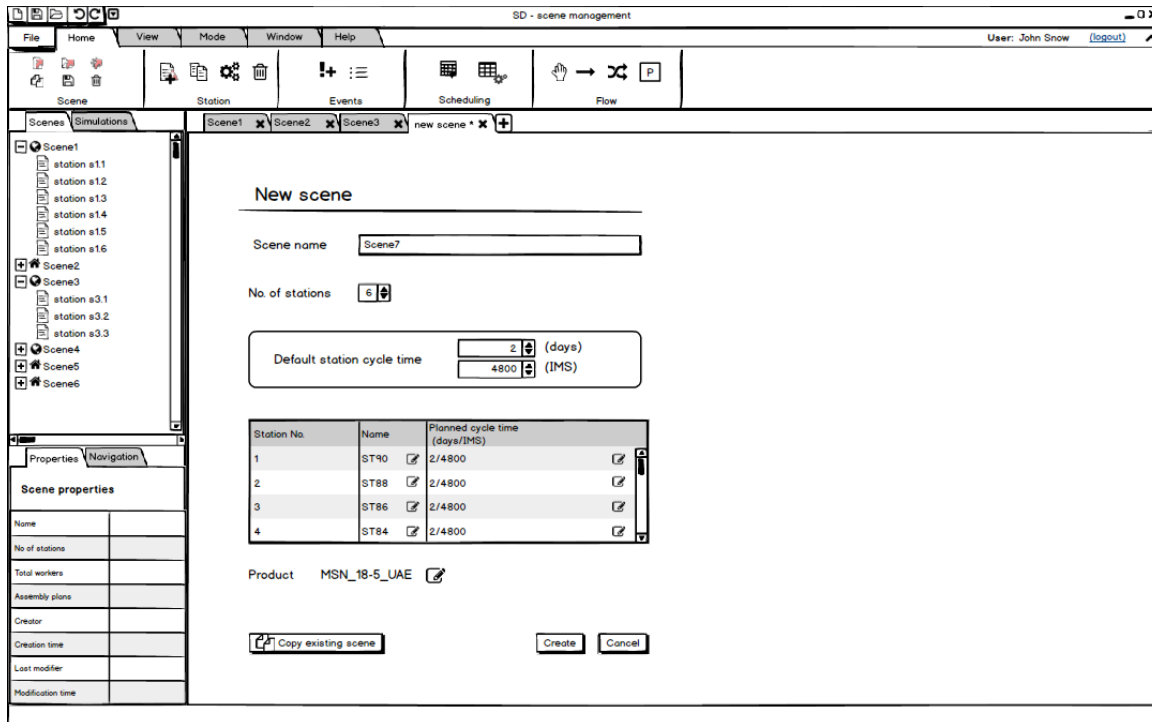


Figure 3.29: Scenario designer - new scene setup

The following section describes the Station setup wizard which is a setup assistant that lead the user through a series of steps (dialogue windows) in order to set up a station. The following mock-ups illustrate the station setup wizard functionality.

### 3.3.2.9 Station setup wizard - general properties

The first step in the Station setup wizard is the general properties setup (figure 3.30), where the user can set the station name and the planned cycle time (both, days and Industrial minutes are supported )If the default scene cycle time is set, the editable cycle time “numeric stepper” is pre-populated.

- The “next button” takes the user to the plans assignment dialog.
- The “cancel button” closes the wizard and no changes are applied to the station setup. Note: the general station properties dialog is opened only if the scene has not yet been set up through the "new scene" component.

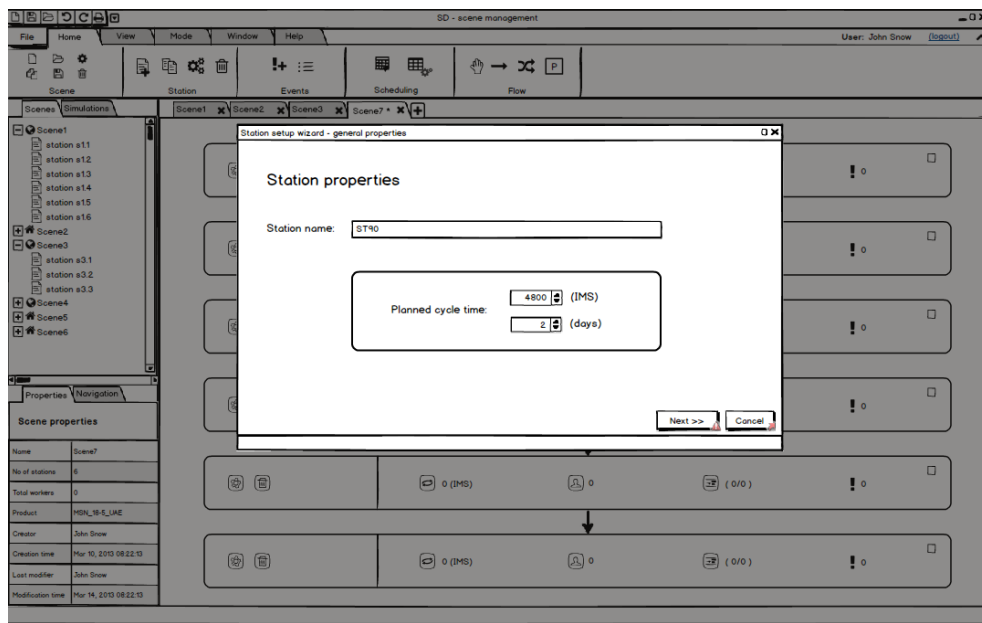


Figure 3.30: Station setup wizard - general properties

### 3.3.2.10 Station setup wizard - assembly plans assignment

The assembly plans assignment dialogue (figure 3.31) displays the work orders assignment component from design mode. The component allows the user to assign a sub-set of the assembly plans to the stations. This component is a part of the FND[6] and therefore, is not described in this document (for further description, see Ondrej Harcuba master thesis[6]).

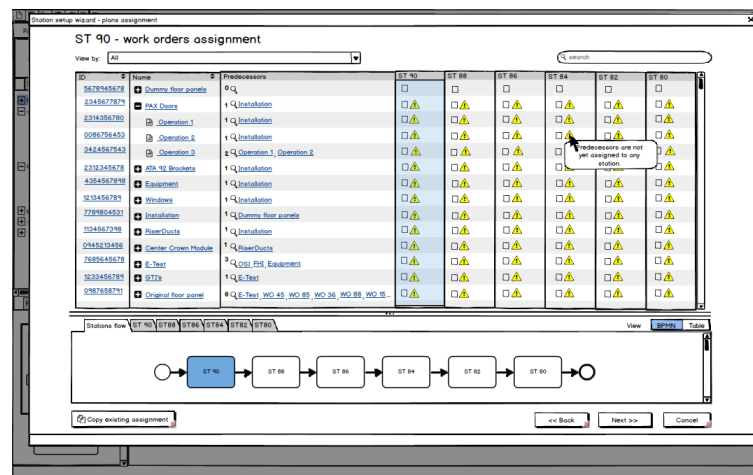


Figure 3.31: Station setup wizard - assembly plans assignment

### 3.3.2.11 Station setup wizard – human resources setup

The next step in the station setup wizard is the human resources setup (see figure 3.32). Here, only generic workers (placeholders) are assigned to a station. The real workers availability is not taken into account. The dialogue description follows:

Displayed skills are static – loaded at the start of the application and the dialog always displays the complete set of existing skills (regardless of the required skills defined in the assigned assembly plans).

Workers are displayed in the first table row – header row. Displayed workers are not static – a worker can be added by pressing the add worker ("+") button right to the last added worker. The add worker button is always visible – pressing the button adds a worker (column) to the human resources table.

The user adds a skill-worker mapping by clicking in the corresponding cell. The warning icon right to skills MAH, MAL, MAP indicates, that according to the assigned as-sembly plans, these skills are required but are not yet assigned.

There is a “view filter” on the right top corner of the human resources table – this component allows the user to filter the displayed data. The following options are available:

- All – displays the entire table data – all skills and workers
- Assigned – displays only the assigned skills
- Unassigned – displays the unassigned skills

A collapsible component with general information is displayed below the human resources setup table. It contains information about the total number of assigned workers, number of required skills, number of assigned skills, number of skills that have to be assigned. If the user decides to collapse the general info component, the human resources table is enlarged.

2 kinds of tooltips are available:

- After hovering the cursor over a skill – displays the skill type and its description.
- After hovering the cursor over a worker – displays the workers competence (sum of all as-signed skills).

The back button leads to the previous setup step – work orders assignment. The next button leads to the next setup step – tools setup. The cancel button closes the wizard and no changes are applied.

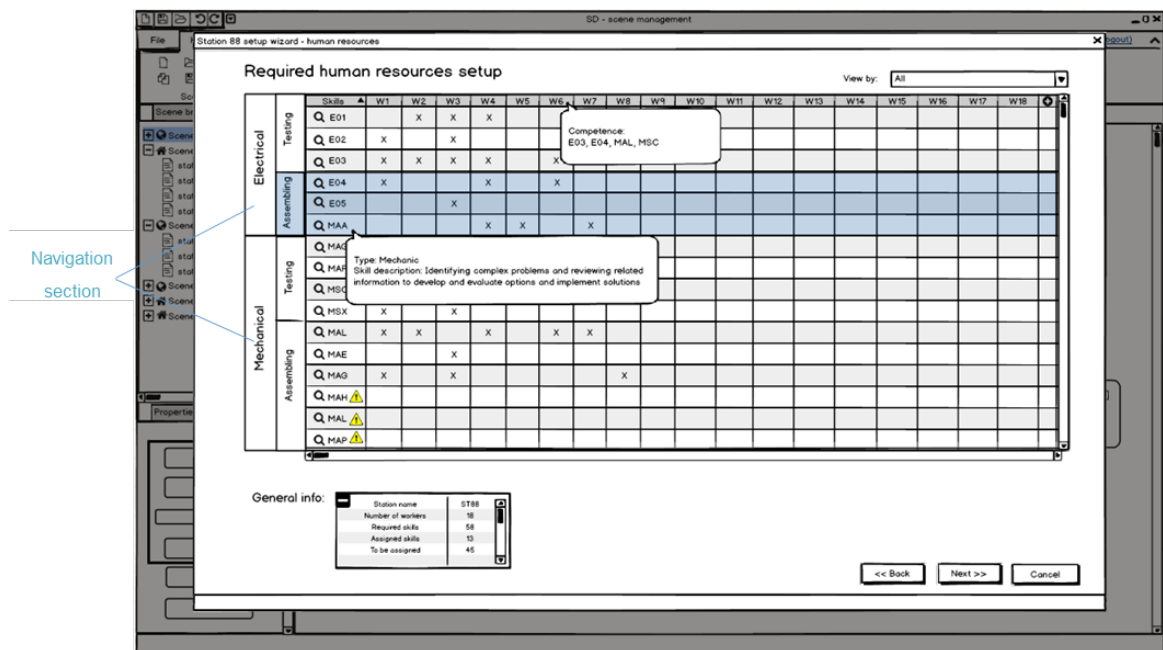


Figure 3.32: Station setup wizard – Human resources setup

### 3.3.2.12 Station setup wizard - tools setup

The last step of the station setup wizard is the tools setup (figure 3.33). The user can view the tools currently assigned to the station and modify their amount, optionally he can add new tools from the tools catalog.

The tools table description follows (columns from left to right):

- The Delete column contains the delete buttons – deletes the tool assignment to the station.
- The Modify column contains the amount modification buttons – pressing the “plus” button increments the tool amount assigned to the station by one, pressing the “minus” button decrements the amount by one.
- Name column (not editable) – contains the name of the tool
- Model column (not editable) – contains the tool model
- Available in stock column (not editable) – displays the amount of currently available tools in the stock/warehouse
- Amount column (editable) – contains the number of tools that are assigned to the station (station 88 in this case).
- The user can modify the amount via the modification buttons or manually, typing a number into the amount column)

Below the tools table, two buttons and a text area is displayed:

- Add tools button – opens the tool catalogue.
- Remove selected tools button – the user is able to select multiple tools and remove them from the tools table.
- Tool details information – text area containing detailed information about the selected tool (High low torque air screwdriver tool in the mock-up)

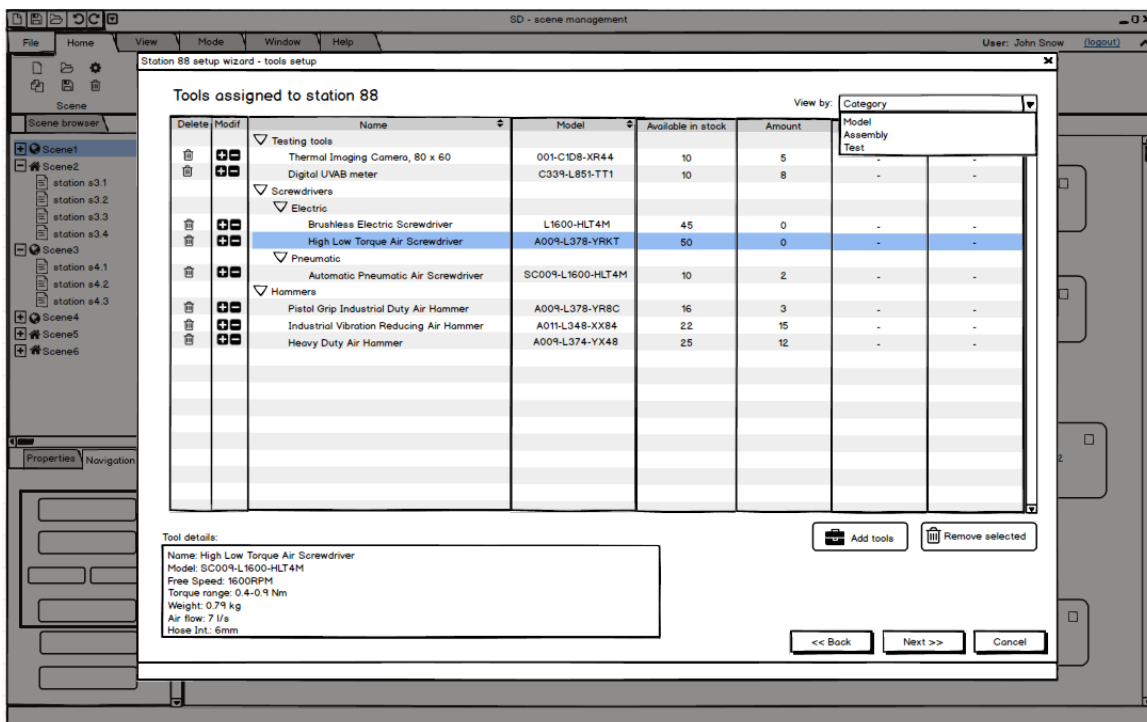


Figure 3.33: Station setup wizard - tools setup



### 3.3.2.13 Station setup wizard - tool catalogue

The tools catalog (figure 3.34) can be invoked from the tools setup dialog in order to assign new tools to a station.

The tool catalog description follows (columns from left to right):

- Select column – contains a checkbox that allows the user to select several tools. If the amount of available tools of a kind is 0, a warning icon is displayed before the checkbox, the checkbox is disabled.
- Name, Model and Available in stock columns are described in section tool setup - (3.34).
- Tool details information below the tools catalog table – is described in section tools setup - (3.34).
- Add selected tools button – adds the selected tools to the tools setup table (figure 3.34).
- Cancel button – returns the user to the tools setup dialog and no changes are applied.

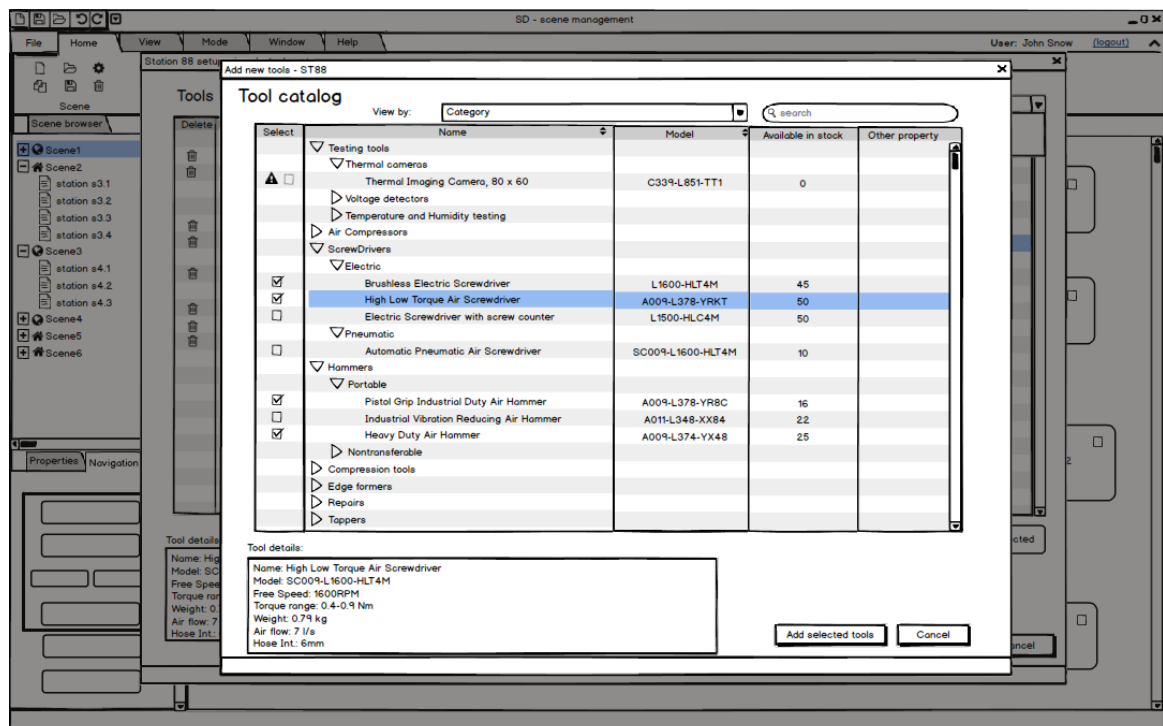


Figure 3.34: Station setup wizard - tool catalogue

### 3.3.2.14 Scheduling properties setup

The scheduling properties setup (figure 3.35) is divided into two sections:

- Left section: allows the user to select the scheduling time range
- Right section: contains general information about the last scheduling

The scheduling properties setup has been simplified (in the figure below is the updated version) in order to facilitate its use to end users.

The screenshot shows the 'SD - scene management' application window. The main workspace is titled 'Generate new schedule' and is divided into two panels. The left panel, 'Re-scheduling properties', contains a 'Scheduling time range' section with four radio button options: 'Current day', 'Current shift' (selected), 'Current cycle', and 'Custom range'. Below these is a 'From' and 'to' date selector. A 'Create new Schedule' button is at the bottom. The right panel, 'Last scheduling', displays a list of scheduling details: 'Created by: John Snow', 'Created: 06:15 13/12/2013', 'Applied: 06:30 13/12/2013', 'Strategy: Corner date', 'Blocked by: MSN\_016-02\_UAE\_13e', and 'Block time: 09:25 13/12/2013'. Below this is a table comparing 'Planned' and 'Current' values for 'Throughput (MSN per week)', 'Corner date', and 'Resource utilization'.

	Planned	Current
Throughput (MSN per week)	2,4	1,9
Corner date	16/12/2013 00:00	16/12/2013 12:00
Resource utilization	78%	92%

Figure 3.35: Scheduling properties setup

### 3.3.2.15 KPIs comparison

Once the result set is available the user is allowed to open the result set and make the KPIs comparison in a new tab (see 3.36).

The KPIs comparison table description follows:

- Delete column – contains a delete button – deletes the result from the simulation.
- Selection column – contains a check box that allows the user to select a result. The selected results can be displayed as a Gantt chart (see figure 3.37). The user can select one, or two results in order to display them in the Gantt chart view.
- Result ID column – displays the result identifier.
- KPIs (Throughput, Corner date, Resource utilization) – displays the results KPIs values. The table can be ordered by a KPI (ascending or descending) by clicking on any of the KPI headers.

The Gantt button below the KPIs comparison table opens the selected results (the schedules) in a new tab (see figure 3.37).

The screenshot shows the 'Simulation1 results - Compare KPIs' table with the following data:

Delete	Selection	Result ID	Throughput (MSN per week)	Corner date (IMS)	Resource utilization
	<input type="checkbox"/>	scene1_V1_res1	3.4	92	80%
	<input type="checkbox"/>	scene1_V1_res2	2.5	229	74%
	<input type="checkbox"/>	scene1_V1_res3	2.2	578	95%
	<input checked="" type="checkbox"/>	scene1_V1_res4	3.1	120	85%

Below the table, there is a 'Gantt' button. To the left, the 'Simulation properties' panel shows:

Name	Simulation1
Number of res.	5
Scene	Scene_L_v10
Assembly plane	A350_B_Lufthansa
Creator	John Snow
Note	the user can add text descri.
Simulation start	Mar 10, 2013 08:22:13
Simulation finish	Mar 10, 2013 08:40:43

Figure 3.36: Scenario designer - KPIs comparison

### 3.3.2.16 Schedules graphical comparison

After the KPIs comparison (see figure 3.36), the user selects one or two results in order to compare the Gantt charts (figure 3.37).

The Gantt charts comparison is performed through the Gantt chart visualizer, which has been developed by my colleague Peter Mathia within his bachelor's thesis[7].

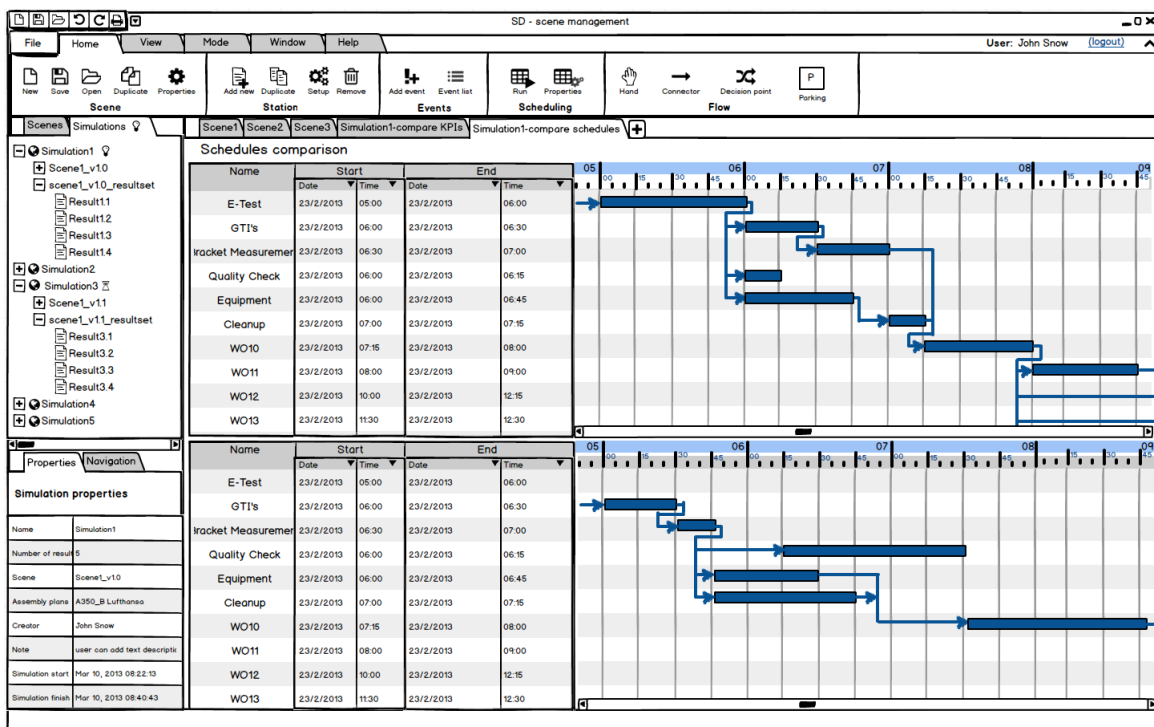


Figure 3.37: Scenario designer - KPIs comparison

### 3.3.3 Production mode

This section provides a brief description of the FNSD production monitoring component - the production mode. A bigger part of this component has been developed by my colleague Ondrej Harcuba[6]. However, in order to fulfill the thesis requirements, I have contributed as well with some components for this mode (e.g. rescheduling properties setup, event insertion, etc.).

### 3.3.3.1 Running production monitoring - stations overview

The production mode stations overview provides the most crucial information regarding the current production state.

Every station is highlighted according to the production manager that is currently viewing the component (e.g. stations 90 and 88 are highlighted to the current station manager, since he is in charge of these 2).

Every station displays the name of the responsible manager, the station name, the current production status (e.g. in progress, blocked state, etc.) and a brief view of the current production progress (tasks done, planned, blocked, etc.)

Every station contains a link (on the right side) to zoom-in into a deeper view of the station status.

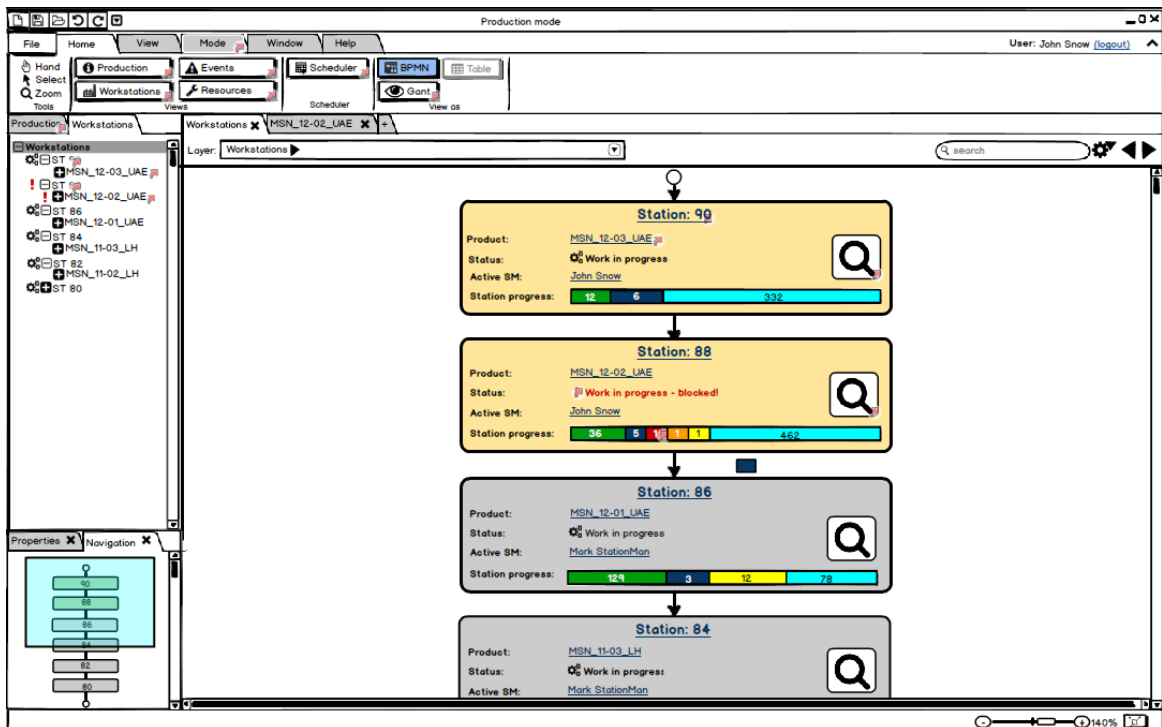


Figure 3.38: Production mode - stations overview

### 3.3.3.2 Running production monitoring - dashboard

The production mode dashboard is the first view provided to the production manager after he logs-in into the FNSD - production mode. The dashboard contains the most important information and provides a quick overview of the current production state.

The production dashboard displays multiple customizable widgets, which allow the user to adapt the user interface to his needs and preferences. Every widget serves also as a link

to a more detailed view of the corresponding type of information (e.g. Gantt chart widget leads to a full view of the Gantt chart visualizer).

In the picture below (figure 3.39), the dashboard contains daily, cycle and performance reports. The Gantt chart widget is displayed on the right side of the production dashboard. The Gantt chart visualizer provides a view of the scheduled tasks in time.

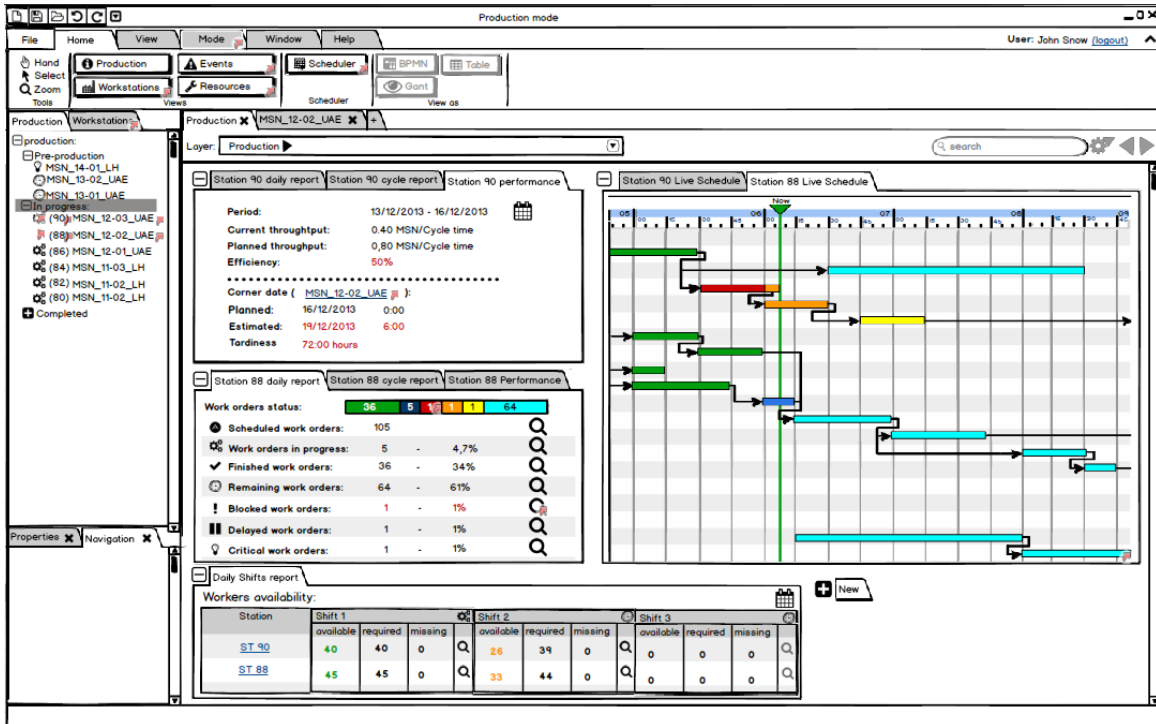


Figure 3.39: Production mode - dashboard

## 3.4 Technology used for the implementation

This section contains a brief description of the technologies used for the implementation of the FNSD.

The FNSD GUI is being implemented as a rich-client component using the JavaFX framework and the Netbeans platform.

The decision of implementing the FNSD interface using this technology was made by ARUM technical leaders. The main reason is the necessity of high levels of responsiveness and stability while processing and displaying large amounts of data, which cannot be achieved by using web-based technologies. Using a rich client application will also allow the user to save data locally and the risk of data loss will be minimized.

Currently, the SD high fidelity prototype is being developed as a standalone application using the JavaFX framework. However, it will be integrated with the FND[6], which is being developed on the netbeans platform. Other ARUM components (e.g. Gantt chart visualizer [7]) are already implemented using the JavaFX framework and the integration with the FNSD will therefore be facilitated.

### 3.4.1 JavaFX

"Since the JavaFX library is written as a Java API, JavaFX application code can reference APIs from any Java library. For example, JavaFX applications can use Java API libraries to access native system capabilities and connect to server-based middleware applications. The look and feel of JavaFX applications can be customized. Cascading Style Sheets (CSS) separate appearance and style from implementation so that developers can concentrate on coding. Graphic designers can easily customize the appearance and style of the application through the CSS "[2].

Some of the JavaFX key features, that have been utilized during the development of the high fidelity prototypes are:

- FXML and Scene Builder. FXML is an XML-based declarative markup language for constructing a JavaFX application user interface. A designer can code in FXML or use JavaFX Scene Builder to interactively design the graphical user interface (GUI). Scene Builder generates FXML markup that can be ported to an IDE where a developer can add the business logic.
- Built-in UI controls and CSS. JavaFX provides all the major UI controls that are required to develop a full-featured application. Components can be skinned with standard Web technologies such as CSS. The DatePicker and TreeTableView UI controls are now available with the JavaFX 8 release.
- Canvas API. The Canvas API enables drawing directly within an area of the JavaFX scene that consists of one graphical element (node).[2]

## 3.5 High fidelity prototypes implementation

This chapter provides the description of the high fidelity prototype of the Scenario designer. The prototype has been implemented using the JavaFX framework, in order to facilitate the integration with the FNSD and other ARUM components being developed in parallel.

The SD high fidelity prototypes (as well as the FND) will allow us to continue with further usability testing. So far, low fidelity prototypes have been tested with end users. The usability tests provided us a good insight into the production processes, and how they are (in reality) being implemented. The obtained feedback revealed a significant amount of unknown information and required changes that will be taken into account in the next implementation phases.

The high fidelity(Hi-fi) prototypes will simulate the FNSD functionality in order to test the GUI design usability in detail. Unlike the low fidelity prototypes, the hi-fi prototypes will simulate the complete functionality defined by the FNSD requirements. However, communication with other ARUM components (e.g. event simulator, FNSD service, MIDAS, etc.) will not be supported yet. The integration with other ARUM components will be performed in later project phases. Many of these components are still being specified and developed.

A brief overview of the SD hi-fi prototypes is provided in the following sections.

### 3.5.1 Scene management

The Scenario designer main view - scene management (figure 3.40) - has been implemented according to the low fidelity design. For further description of the component's functionality, see section 3.3.2.1.



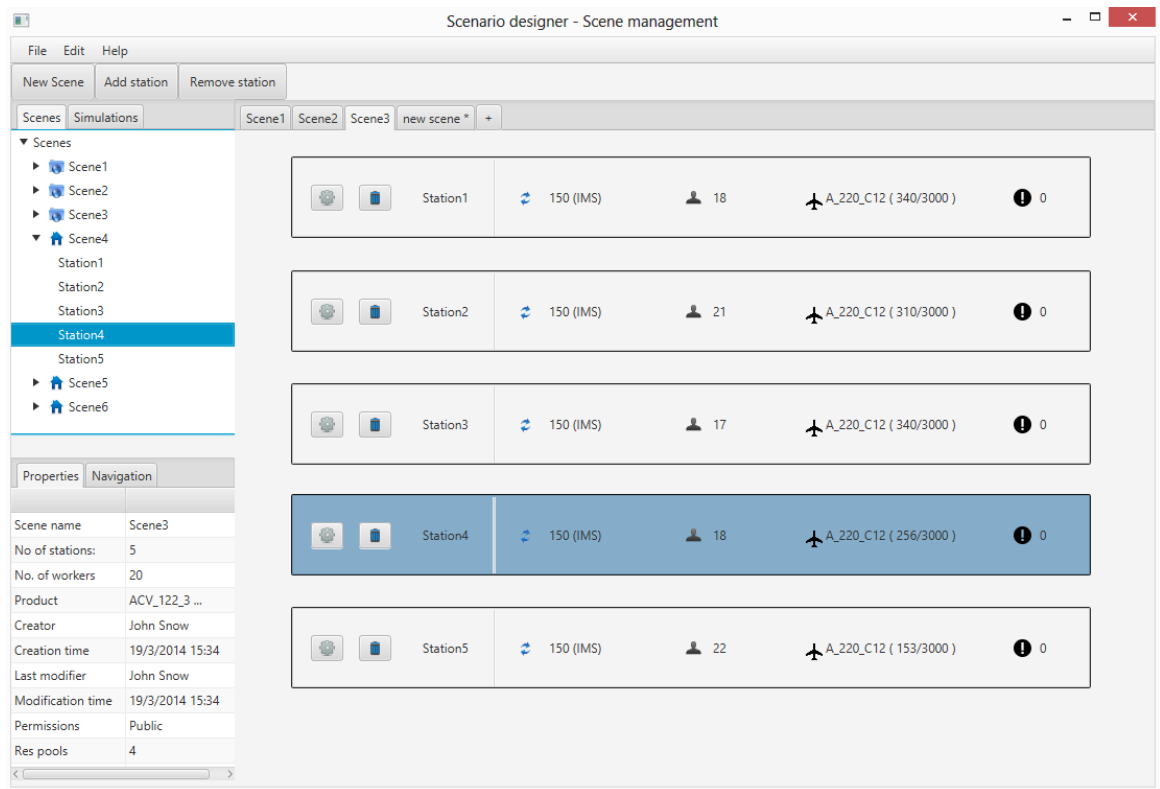


Figure 3.40: SD high fidelity prototype - scene management

### 3.5.2 New scene

The new scene form (figure 3.41) - has been implemented according to the low fidelity design. For further description of the component's functionality, see section 3.3.2.8.

### 3.5.3 Station setup wizard - human resources setup

The human resources setup dialogue (figure 3.42) - has been implemented according to the low fidelity design. For further description of the component's functionality, see section 3.3.2.11.

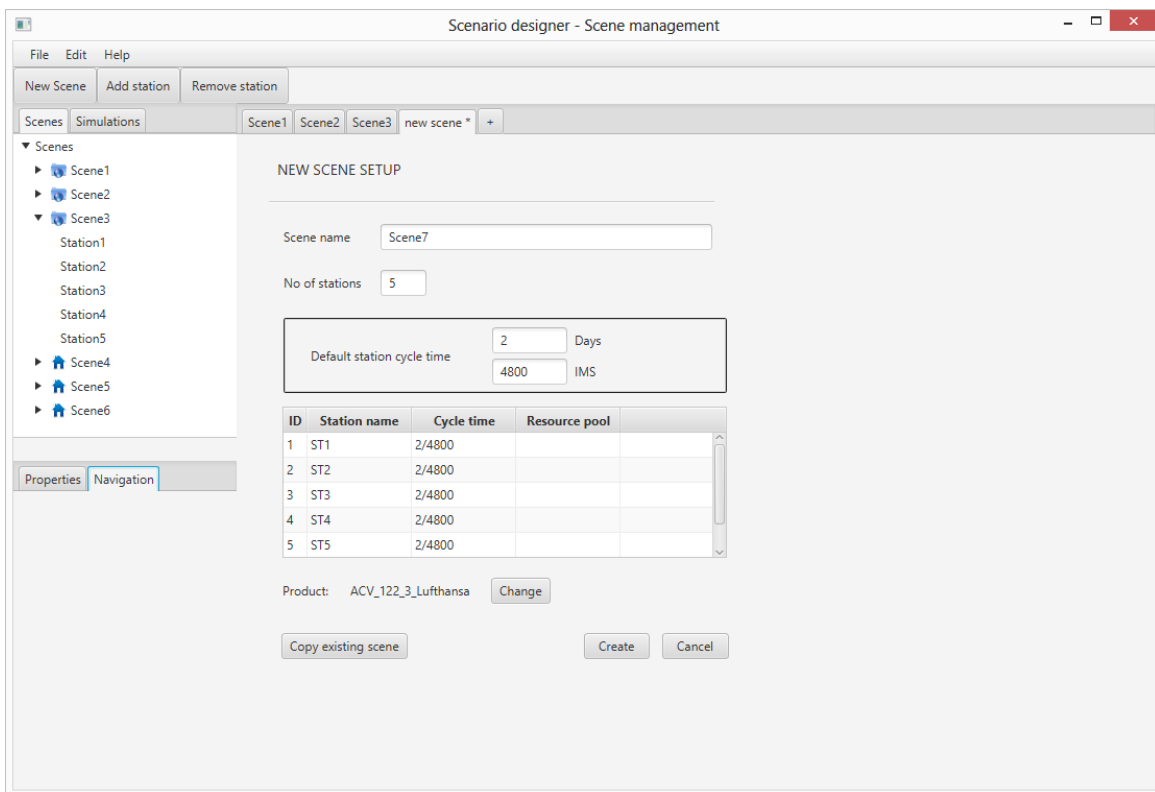


Figure 3.41: SD high fidelity prototype - scene management

Station setup wizard - Human resources

**Human resources setup - station3** View by: All

		Skill	w1	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12	w13	w14	w15	w16	w17	w18	w19	w20		
<b>Electrical</b>	Testing	E01		X	X			X					X				X		X	X				
		E02		X		X	X	X				X	X	X			X	X	X					
		E03				X	X		X	X	X	X	X		X	X	X	X		X	X			
		E04	X						X	X								X	X	X	X			
	Assembly	CKT	X		X	X		X	X		X					X		X	X	X				
		CKK	X	X	X	X	X		X	X	X	X	X			X	X		X		X			
		MEM		X	X		X	X	X		X					X		X		X	X			
		RAM	X	X		X	X				X			X		X						X		
RA2	X		X		X	X	X	X	X			X	X	X		X	X	X						
<b>Mechanical</b>	Testing	SA3				X	X		X		X	X	X											
		SR3		X				X	X			X	X	X					X	X	X			
		SR4	X	X								X			X					X	X			
		QE1	X	X	X			X	X		X		X	X	X	X	X	X	X		X			
	Assembly	QE2		X		X	X	X	X	X						X	X	X	X		X			
		YTT		X	X	X					X	X	X			X	X	X	X	X	X			
		YKM	X						X	X		X		X	X		X		X					
		YMM						X			X		X		X		X	X	X	X				

General Info: << Back    Next >>    Cancel

Number of workers	42
Electricians:	15
Mechanics	27
Testers	6

Figure 3.42: SD high fidelity prototype - scene management



## Chapter 4

# Usability testing

This chapter describes the FNSD usability tests, which were held at Airbus facilities in Hamburg, Germany. The first usability test was held in March 2014 and its main purpose was to eliminate major errors in the GUI design. The second testing took place in April 2014. We were able to adapt the low fidelity prototype to the end users from the first testing, and the second revealed more subtle errors.

The following section contain a short sample from the usability testing report.

### 4.1 Usability testing

This section contains a report from the FNSD usability testing held in Hamburg, Germany - April 2014.

#### 4.1.1 Purpose of the usability testing and expected output

The purpose of the user interface usability testing at 16th of April in Hamburg was to obtain user feedback on the FNSD graphical interface (so far as an interactive prototype). The feedback was obtained from several potential users / experts who represent primarily people working and organizing work at the assembly stations (Station managers, Production steering).

The experts were asked about their daily jobs and major problems they face. A scenario was introduced (appendix A) and they were asked to go through the interactive prototype and to answer questions like:

- Is the information presented easy to understand?
- Do you know what individual items represent and what do you expect the controls would do?
- Did the controls do what you expected?
- Does the tool cover your needs?
- Is the accessibility of the user interface satisfactory?

The following section contains a short sample of the report from the usability tests.

### 4.1.2 Scheduling properties

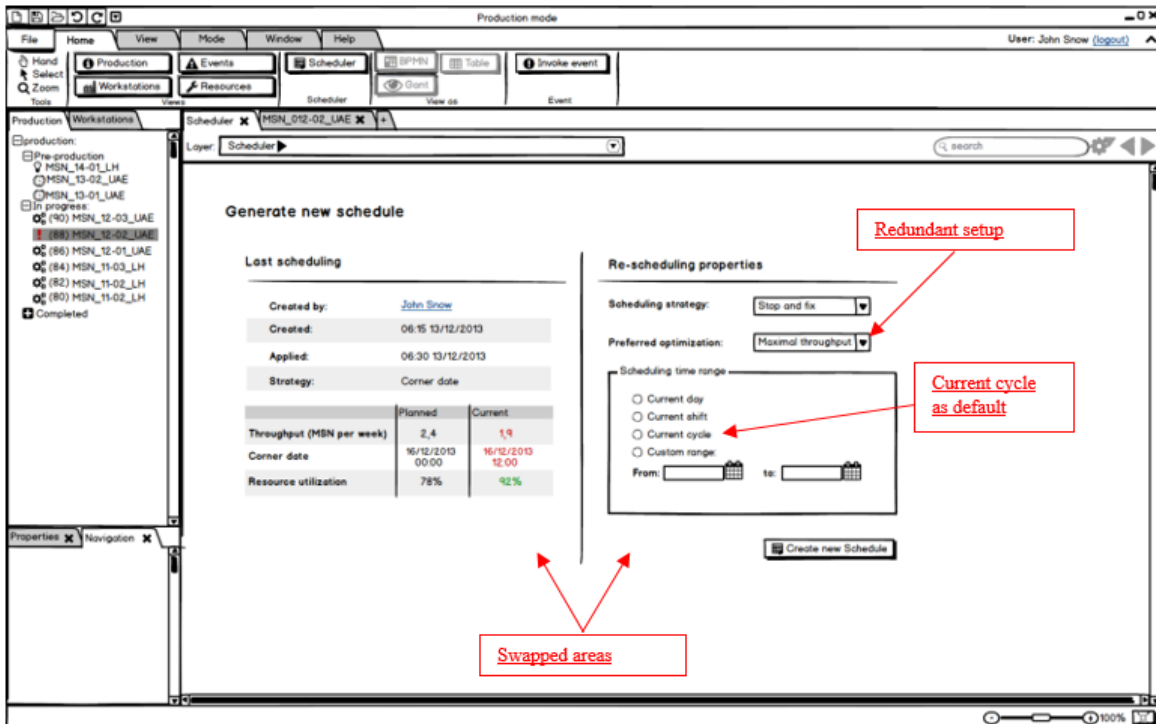


Figure 4.1: Rescheduling - reported issues

#### 4.1.2.1 Issue: Redundant setup – preferred optimization

The user suggested to simplify the scheduling properties setup. Moreover, setting up the preferred optimization prior to the scheduling is erroneous.

**Recommendation** Remove the “preferred optimization” option.

#### 4.1.2.2 Issue: Scheduling time range – current cycle as default

The user would prefer to have the “current cycle” time range as default. He also mentioned that “current day” and “current shift” could be useful as well.

The user asked about whether it would be possible to obtain a “recommended time range” from the application for every rescheduling. This issue will be further discussed.

**Recommendation** Set the “current cycle” scheduling time range as default. Consider the possibility of providing the user a “recommended time range” for every rescheduling (based on the current situation).

#### 4.1.2.3 Issue: Scheduling properties – swapped areas

Area “last scheduling properties” should be swapped with area “re-scheduling properties”. Most important information should be displayed on the left side on the screen.

**Recommendation** Swap area “last scheduling properties” with area “re-scheduling properties”.

#### 4.1.3 Results comparison

The screenshot shows a software interface with a 'Compare KPIs' table. The table has columns for 'Delete', 'Selection', 'Result ID', 'Throughput (MSN per week)', 'Corner date (IMS)', and 'Resource utilization'. The data rows are as follows:

Delete	Selection	Result ID	Throughput (MSN per week)	Corner date (IMS)	Resource utilization
<input type="checkbox"/>	<input type="checkbox"/>	13122013	3.4	+ 92	80%
<input type="checkbox"/>	<input type="checkbox"/>	13122014	2.5	- 229	74%
<input type="checkbox"/>	<input type="checkbox"/>	13122015	2.2	- 578	95%
<input type="checkbox"/>	<input type="checkbox"/>	13122016	3.1	+ 120	85%

Annotations in the image point to the following issues:

- Add new KPI – amount of travelling work**: Points to the 'Throughput (MSN per week)' column header.
- Change throughput unit**: Points to the 'Throughput (MSN per week)' column header.
- Erroneous abbreviation**: Points to the 'Corner date (IMS)' column header.

Figure 4.2: Results comparison - reported issues

##### 4.1.3.1 Issue: Erroneous industrial minutes abbreviation

The user could not recognise the abbreviation used for “industrial minutes – IMS” in the prototype.

**Recommendation** Change IMS to IM in order to match the currently used abbreviation.

##### 4.1.3.2 Issue: Add new KPI – amount of travelling work

The conversation with the user revealed the requirement of adding a new KPI – “amount of travelling work”.

**Recommendation** Add a new column in the “scheduling results” table with the new KPI – amount of travelling work.

#### 4.1.3.3 Issue: Change throughput unit

The current throughput unit (MSN/week) represents a relatively short time range (the cycle time under ramp up is about 8 days).

**Recommendation** Change the throughput unit to MSN/month.



# Chapter 5

## Conclusion

### 5.1 Thesis summary

The main goal of this thesis was to develop the Scenario designer: a managerial tool within the ARUM project to support monitoring, visualization and scheduling of aircraft production in ramp-up.

This thesis contains the functional requirements, functional specification, the architecture design, the graphical user interface design and the implementation (as a high fidelity prototype) of the Scenario designer.

At the date of this thesis submission, the architecture design and the mock-up prototypes of the Scenario designer and the Factory network designer (together referred as FNSD) have been developed into such stage, that allowed us to start with the implementation phase.

A sub-set of the FNSD design has been tested with end users at Airbus facilities (March and April 2014). A significant amount of valuable feedback was retrieved from both usability tests, which will be taken into account during the following implementation phases.

#### 5.1.1 Further work

The Scenario designer, in its final stages, will be integrated with other ARUM tools developed in parallel within the project. Nowadays, many of the tools are not yet finished and cannot be connected to or used by the SD. One of the tools to be used (its sub-functionality) by the SD is the *Factory network designer* [6], which is being developed by my colleague Ondřej Hrcuba, within his master's degree thesis.

As previously mentioned, only low fidelity and high fidelity prototypes (a sub-set of the overall functionality) could have been implemented so far. Nevertheless, the ARUM project naturally continues and after the submission of this thesis, the development of the Scenario designer will be finished.



# Bibliography

- [1] *ARUM: Adaptive Production Management* [online]. Dostupné z: <http://arum-project.eu/index.php/en/overview>.
- [2] *JavaFX Overview* [online]. Dostupné z: <http://docs.oracle.com/javase/8/javafx/>.
- [3] BIELE, A. D2.1.1 - Use-case Definition (use-cases 1&2). Status: confidential, 2013.
- [4] BUDIN, T. ARUM Development guidelines. Status: confidential, 2013.
- [5] GLEICH, C. F. – SCHÜTT, A. – ISENBERG, R. SMART ramp-up: methods to secure production ramp-up in the aircraft industry. *CEAS Aeronautical Journal*. 2012, 3, s. 125–134. doi: 10.1007/s13272-012-0047-7.
- [6] HARCUBA, O. Tool for operational monitoring of production processes. Czech Technical University in Prague, 2014.
- [7] MATHIA, P. Tool for decision support in production planning and scheduling. Czech Technical University in Prague, 2014.
- [8] MÜLLER, D. – ANHALT, A. D2.2.1 - use case detailing and KPI setting. Status: confidential, 2013.
- [9] PARTNERS, A. D4.1.1 System architecture and platform specifications. Status: confidential, 2013.
- [10] PARTNERS, A. Description of work. Status: public, 2012.



# Appendix A

## Glossary

Name	Definition
<b>Application mode</b>	Graphical interface unit designed to fulfil a specific use case. This document contains description of the Scenario designer mode.
<b>ARUM</b>	Adaptive production management
<b>ARUM computational core</b>	Computing unit designed to perform the WOs/jobs scheduling (also called scheduler). Transforms an input (scene) into one or multiple outputs (schedules).
<b>Assembly plan</b>	Graphical representation of all work orders (operations) of a product and their technological interrelations.
<b>AIB</b>	Airbus
<b>DESC</b>	Description
<b>DEP</b>	Dependency
<b>ESB</b>	Enterprise service bus: software architecture that allows the communication between multiple application modules in a service-oriented architecture.
<b>FND</b>	<b>Factory network designer</b> : a graphical user interface developed within the ARUM project. Functionality: management of assembly plans and current production monitoring.
<b>FNSD</b>	The top-level application that contains multiple modes developed within the ARUM project. The modes are: Design mode, Production mode, Scenario mode.

<b>Graphical interface component</b>	An independent graphical interface sub-unit designed to provide specific functionality. A graphical component provides always the same functionality regardless the current application mode or the current view (might display different type of data).
<b>ICT</b>	Information communication technology
<b>KPI</b>	Key performance indicator
<b>Live scene</b>	The scene that represents the running production state.
<b>MR</b>	Missing resource event
<b>NC</b>	<b>Non-conformity:</b> a defect in an item that substantially prevents the item or service from operating or functioning as designed or according to its specifications. A nonconformity is a severe deviation from a specification. A nonconformity has to be approved by a station manager, after it has been reported by a worker.
<b>Production line</b>	An organized factory system. Components of the end product are assembled in a number of different stations at each of which a manual and/or machine operation is performed. The production line consists of stations. A product might (AIB case) pass through all stations, or might remain in one station (IHF case) the whole production lifecycle.
<b>SD</b>	<b>Scenario designer:</b> a graphical user interface developed within the ARUM project. Functionality: management of scenes and current production monitoring.
<b>Scene</b>	An abstract entity representing the factory properties and state. See scene definition in scene lifecycle.
<b>Station</b>	A self-contained workspace equipped with workers, tools, kittings. A station has defined cycle time, assigned product(s) and flow in context of other stations. A station is part of the scene(see SCENE LIFECYCLE).
<b>User</b>	Every person that interacts with the application.
<b>What-if games</b>	Simulations performed through ARUM tools in order to analyse possible production progress.
<b>What-if scenario</b>	An instance of the what-if games.

<b>Work order (WO)</b>	Assembly/net plans consists of work orders. Each work order consists of jobs.
------------------------	---

Table A.1: Glossary





# Appendix A

## Content of attached CD

- text - pdf and "tex" source files of this thesis
- source - source codes of the prototypes



# Appendix A

## Usability testing scenarios

### A.1 Rescheduling scenario

A station manager (you) comes to the morning shift and wants to review the performance of the currently running production at his/her station(s).

#### **Dashboard:**

- Identify the KPIs of the production
  - What is the percentage of finished/remaining work orders/processes?
  - Is the production delayed?
- What is the current situation at your stations?
  - What is the current progress of each station?
  - Which station is blocked? Why?
- How many blocking events there are? How many of them are new?
- What is the current situation regarding the shifts?
  - How many workers are available at ST 90 for shift 1?
  - How many workers are required at ST 88 for shift 2?

#### **Assembly plan/Gantt chart:**

- Which work orders / processes are now running?
- Which work orders / processes are now finished?
- Which work orders / processes are in trouble?
- Which processes (work orders) are being done at my station that did belong to other stations?

- Which processes (work orders) originally planned to be done at my station are delayed and transferred to other stations?
- Display the critical path

### **Resources**

- What workers should be in the current shift for ST 90?
- What work orders / processes are assigned to worker Jackson Whitledge?
- What workers are currently working?
- What workers are available?

### **Events**

A new event occurs – reported by one of the workers.

- Identify what happened – what type of event occurred
- Filter the events – display only the blocking events
- Browse details of a blocking event
- Who reported the event?
- On which station the event occurred?
- When was the event reported?
- Who is currently responsible for solving the event?

### **Rescheduling**

You have decided to create a new schedule, based on the current situation.

- Run the rescheduling
- Setup the scheduling parameters
- Setup the scheduling strategy: traveling work
- Setup the preferred optimization: corner date
- Setup the scheduling time range: current day
- Display the results
- How would you compare the KPIs of the results?
- What is the best result regarding the maximal throughput?
- Compare two Gantt charts
- Apply your selected schedule to the current production