FP7 Project 2007- Grant agreement n°: 218575

Project Acronym: **INESS**

Project Title: **INtegrated European Signalling System**

Instrument: Large-scale integrating project
Thematic Priority: Transport

INESS_WS G_Deliverable G4.2_System and Software Specification

Due date of deliverable	2010-06-30
Actual submission date	2010-10-04

Deliverable ID:	D.G.4.2
Deliverable Title:	System and Software Specification
WP related:	System and Software Specification for Tool Support
Responsible partner:	Funkwerk
Task/Deliverable leader Name:	Carsten Trog
Contributors:	Funkwerk, TUBS, DLR, DB, Banverket, ProRail, BBR, RFI, Ansaldo

Start date of the project: 01-10-2008 Duration: 36 Months

Project coordinator: George Barbu
Project coordinator organisation: UIC

Revision: WS Final Dissemination Level[1]: CO

DISCLAIMER

The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

PROPRIETARY RIGHTS STATEMENT

This document contains information, which is proprietary to the INESS Consortium. Neither this document nor the information contained herein shall be used, duplicated or communicated by any means to any third party, in whole or in parts, except with prior written consent of the INESS consortium.

[1] PU: Public, PP: Restricted to other programme participants (including the Commission Services), RE: Restricted to a group specified by the consortium (including the Commission Services), CO: Confidential, only for members of the consortium (including the Commission Services).

Document Information

Document type: Deliverable
Document Name: INESS_WS G_Final 4.2_Ver2010-10-04
Revision: WS Final
Revision Date: 2010-10-04
Author: Rüdiger Neumann, G. v. Buxhoeveden
Dissemination level: CO

Approvals

	Name	Company	Date	Visa
WP leader	Carsten Trog	Funkwerk IT	04-10-2010	Validated
WS Leader	Carsten Trog	Funkwerk IT	04-10-2010	Validated
Project Manager	Emmanuel Buseyne	UIC	26-10-2010	Validated
Steering Board				

Document history

Revision	Date	Modification	Author
1	2009-07-03	Creation of document	J. Schröder
2	2009-07-24	SRS, SWRS added	G. Buxhoeveden
3	2009-07-31	Sections for Evaluation, System Architecture, GSN completed and List of functions in Annex added	G. Buxhoeveden
4	2009-07-31	Revised and -structured, GSN-Introduction shifted to D.G.2.2	J. R. Müller
5	2010-06-01	System- and Software Requirenments	R. Neumann
6	2010-06-04	Overall revision, adjustment to other Deliverables	G. Buxhoeveden
7	2010-07-16	Finalising, no content changes	C. Trog
8	2010-10-04	Revision of the System- and Software-Spezification	R. Neumann
WS Final	2010-10-25	Format and quality checking	Richard Vaux,PMO

TABLE OF CONTENTS

Glossary

The following abbreviations are applied in this document:

BPM	Business Process Modell
CIFS	Common Internet File System (extension of SMB)
DB	Database
DMS	Document Management System
GSN	Goal Structuring Notation
JRE	Java Runtime Environment
SMB	Server Message Block
WebDAV	Web-based Distributed Authoring and Versioning
SRS	Software Requirement Specification

Section 1 – Executive Summary

1.1 The context of workstream G

The aim of workstream G is to reduce time and money for the Safety Case in industry, i.e. operators as well as suppliers, by avoiding unnecessary or redundant procedures. To achieve this aim one can identify four phases in workstream G (see figure 1).

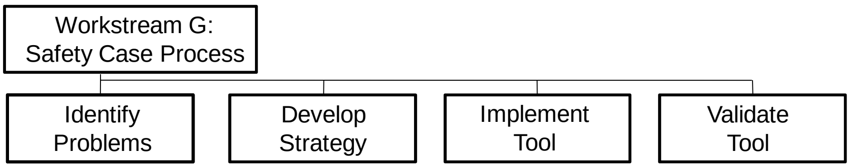


Figure 1: One can specify four phases to achieve the aim of Workstream G

According to the DoW for the second and third phase the following procedure was planned:

1. Define long term and short term goals (task 2.1 and task 2.2)
2. Estimate economical benefit (task 2.3)
3. Specify requirement description technique (task 3.2)
4. Specification of improved Safety Case Process (task 4.1)
5. Specification of system and software requirements (task 4.2)
6. Implementation of tools (task 5.1)

1.2 The aim of task G.4.2

The output from this task shall be a system and software requirements specification, that describes the tool functions to implement and the system (hardware and operating system) on that the software shall be used.

All functions of the future tool have to be described in detail in the requirement specification. So the functions of the tool can be agreed upon before starting programming. The programmer then has an exact specification of what he is supposed to program, which reduces redesign, recoding and retesting. Furthermore, validation of software is only possible if the requirements are well defined.

For the requirements specification being useful the requirements have to be unambiguous, consistent, testable and complete. This can be reached easier by the use of formal or semiformal description means and thorough reviews.

Section 2 – System and Software Requirements Specification

2.1 The System Requirements Specification

The means of description, methods and tools to describe the requirements are specified in D.G.3.2

Each requirement shall have an individual identifier that can be used to refer to it.

The following scheme is used:

R<source-shortcut>_<consecutive number>

"R" is used for Requirement. The following shortcuts shall be four letters big.

Source can be:

- SYST -> System Requirement
- DMST -> Document management system Requirement
- WFLT -> Workflow-Tool Requirement
- GSNT -> GSN-Tool Requirement

If a requirement realizes another requirement, this should be indicated appropriately. If lower level documents are needed, the apportionment of the requirements to the respective documents should be made obvious, e. g. by a forwarding table.

2.1.1 The List of System Requirements

Req-ID	RSYST_001
Header	Open source SW-Tool as basis for SaCaPro
Description	One or several open source software tools shall be used as a basis for the development of the SaCaPro.
Comment	using of the basis tools, makes it easier to manage the CENELEC-process
Partly realizes ReqID	

Req-ID	RSYST_002
Header	Basic tools: Extensible
Description	The basic tool(s) shall be extensible.
Comment	
Partly realizes ReqID	

Req-ID	RSYST_003
Header	Basic tools: Operation System Independence
Description	The basic tool(s) shall run on different Operating Systems.
Comment	
Partly realizes ReqID	

Req-ID	RSYST_004
Header	Basis tools: Database System independence
Description	The basic tool(s) shall be able to communicate with different Database Systems.
Comment	
Partly realizes ReqID	

Req-ID	RSYST_005
Header	Basic tools: modular
Description	The basic tool(s) shall be modular (and therefore exchangeable)
Comment	
Partly realizes ReqID	

Req-ID	RSYST_006
Header	Basic tools: Ability to communicate with existing systems
Description	The basic tool(s) shall be able to communicate with Standard Tools (Microsoft-Office)
Comment	
Partly realizes ReqID	

Req-ID	RSYST_007
Header	Basic tool: Development in V-process
Description	The basic tool(s) shall be support developing within v-process.
Comment	
Partly realizes ReqID	

2.2 The Software Requirements

2.2.1 The List of Software Requirements

Req-ID	RDMST_001
Header	DMS: Open communication interface
Description	The DMS must offer at least one open Communication Interface
Comment	
Partly realizes ReqID	

Req-ID	RDMST_003
Header	DMS: Single place for all Documents
Description	The DMS shall provide a single place for all documents that are related to the SaCa.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_004
Header	DMS: Upload
Description	All Documents that are available as a file can be managed with the DMS.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_005
Header	DMS: Integration of Office-tools
Description	The DMS shall have integration for Office-Tools. At least (MS-Office, OpenOffice, Visio, AcrobatReader/Writer) to view and edit the uploaded documents
Comment	
Partly realizes ReqID	

Req-ID	RDMST_006
Header	DMS: Backup-facilities
Description	The DMS must provide backup-facilities to avoid loss of data
Comment	
Partly realizes ReqID	

Req-ID	RDMST_007
Header	DMS: Scheduling meetings
Description	The DMS shall provide a tool to schedule meetings
Comment	
Partly realizes ReqID	

Req-ID	RDMST_009
Header	DMS: Bulletin board system
Description	The DMS shall integrate tools for information exchange (Bulletin board System)
Comment	
Partly realizes ReqID	

Req-ID	RDMST_010
Header	DMS: Versioning
Description	The version number of documents in the DMS must be increased automatically , when a document is changed. The Document itself keeps his version number. This is only changed by the user.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_011
Header	DMS: Tagging
Description	It must be possible to set tags to the documents. (e.g. 'safety relevant')
Comment	
Partly realizes ReqID	

Req-ID	RDMST_012
Header	DMS: State
Description	It must be possible to set states to the documents
Comment	
Partly realizes ReqID	

Req-ID	RDMST_013
Header	DMS: Rules to set states
Description	It must be possible to define rules to set states. The transitions must be dependent on the role of the user

Comment	
Partly realizes ReqID	

Req-ID	RDMST_014
Header	DMS: Define users and roles
Description	It must be possible to administrate Users and roles of the users.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_015
Header	DMS: Notification
Description	The Notification can be done inside the DMS or via e-mail.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_017
Header	DMS: Link functions; general
Description	The DMS must have a function to link documents. Links to documents that are based on the type of document shall be used to realize the peculiarities of the SaCa
Comment	
Partly realizes ReqID	

Req-ID	RDMST_018
Header	DMS: Link functions; between any documents
Description	link between any documents (DMS standard): Any two documents can be linked to each other. The DMS shall take care that all parties involved in document creation are informed about changes of the document.
Comment	
Partly realizes ReqID	

Req-ID	RDMST_024
Header	DMS: Search functions; meta-information
Description	Search within document meta-information
Comment	(e.g. author, version, language, tags etc.)
Partly realizes ReqID	

Req-ID	RDMST_025
Header	DMS: Search functions; all documents
Description	Text-Search within all documents
Comment	
Partly realizes ReqID	

Req-ID	RDMST_026
Header	DMS: Search functions; related tp a project
Description	Search within documents related to a project
Comment	
Partly realizes ReqID	

Req-ID	RDMST_027

Header	DMS: Search functions; whole system
Description	Search within the BBS / Knowledge Database / Lessons learned Database / Product + Expert Database / Online Help
Comment	content unclear, a little bit to heavy
Partly realizes ReqID	

Req-ID	RDMST_029
Header	DMS: Automatic Documentlist
Description	Generate document-list from a certain projekt (name, version, state ...)
Comment	
Partly realizes ReqID	

Req-ID	RDMST_030
Header	DMS: Administer multiple Projects/SaCases
Description	In the DMS it shall be possible to manage multiple projects (safety-cases).
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_001
Header	WFT: Standard Workflow-Tool
Description	A standard WF-Tool shall be used unless the DMS-Tool has no WF-functionality
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_002
Header	WFT: Define workflows for DMS
Description	The Workflow-Tool must be able to define workflows suitable for the DMS
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_004
Header	WFT: Status
Description	In an overview the status of all SaCa related documents shall be shown to get a report of missing documents to complete a SaCa.
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_005
Header	WFT: Requirements
Description	Requirements are not only linked to documents of the SaCa as mentioned above. The tool shall provide a checklist that reports the fulfillment of all requirements.
Comment	.
Partly realizes ReqID	

Req-ID	RWFLT_007
Header	WFT: Approval/rejection process
Description	Within the SaCa workflow documents are committed from one party to another. Depending on the workflow, documents can change their state (e.g. from 'not approved' to 'approved') or can be propagated to another stage in the workflow process (e.g. move from 'testing' to 'verification').
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_011
Header	WFT: Document state change
Description	A document changes its state, once a user has performed a certain action with the document (e.g. a user has approved a document).
Comment	
Partly realizes ReqID	

Req-ID	RWFLT_013
Header	WFT: Grapical user interface/transition-diagramm
Description	The workflow tool shall provide a graphical user interface to define transition-diagramm graphically.
Comment	we have to define what a workflow is and how it works. For one document you have the state, the roles and the users. Changing the state of a document is depending from these three things. .
Partly realizes ReqID	

Req-ID	RWFLT_014
Header	WFT: user interface/defining roles
Description	The workflow tool shall provide a user interface to define the roles for every transition.
Comment	we have to define what a workflow is and how it works. For one document you have the state, the roles and the users. Changing the state of a document is depending from these three things. .
Partly realizes ReqID	

Req-ID	RWFLT_015
Header	WFT: user interface/notations
Description	The workflow tool shall provide a user interface to define notation on every transition??
Comment	we have to define what a workflow is and how it works. For one document you have the state, the roles and the users. Changing the state of a document is depending from these three things. .
Partly realizes ReqID	

Req-ID	RGSNT_001
Header	Interface GSN/WFT - DMS

Description	The GSN Tool must be able to interface with the DMS
Comment	
Partly realizes ReqID	

Req-ID	RGSNT_002
Header	GSN document-templates
Description	Create Directory-Structure with document-templates from templates
Comment	
Partly realizes ReqID	

2.3 Evaluation of existing Open Source Software

To provide an overview of the Tool to be realized, a top-down approach is a helpful measure to describe the overall design. Therefore, in the sections to follow a general layout in the form of a *System Architecture* is described. The scope of the architecture is to put as many open source-"of the shelf" and "exchangeable" tools together to stay as flexible as possible in the later process of choosing already available packages that conform to the architecture. I.e. the tool to be delivered will consists of several exchangeable tools that are at least partly integrated. The degree of implementation will be decided after the evaluation of the open source tools has been completed.

Deliverable D.G.2.1 (Long Term Goals) describes the rationale behind the evaluation process of the open-source tools chosen. As a result an open-source Document Management System was chosen to be the file storage system for the implementation of further tools. Additionally DMS conforms to open communication standards, which facilitate the modular integration of such tools.

In the section about *Programming and Configuration* it becomes clear, why we chose the very overall system architecture: Only a small part of the system needs to be configured (workflows, users, groups etc. within the DMS), another tool (the GSN tool) needs to be programmed to talk to existing or customized interfaces of the DMS.

The principles of a *Workflow Tool* are presented to show how they integrate with the DMS.

The *GSN Tool* is described as well to get an idea, how the system will work.

Finally an overall *User Experience* is shown.

2.3.1 System Architecture

In the second WS G workshop on June 2009 it was decided, to use existing Open Source components to facilitate and ease the development of a Safety Case Tool. It was realized that many, if not most functions, that were identified in user interviews and in workshop discussions could be achieved with a DMS.

To identify the most suitable DMS the above mentioned System Requirements (see 2.1) were taken as a basis to evaluate existing solutions.

A generic System Architecture, which depicts the System Requirements, is shown in Figure (A). The whole system is client-server based. Most functions are based on exchangeable standard components (Operating System, Java Runtime Environment, Web Browser, Office Software, Workflow Tool).

A DMS usually consist of components which are interrelated (i.e. they will work with a few different Databases, but not with every Database, they might run on a certain application server, but not on another, they accept a certain kind of workflow format, but not another).

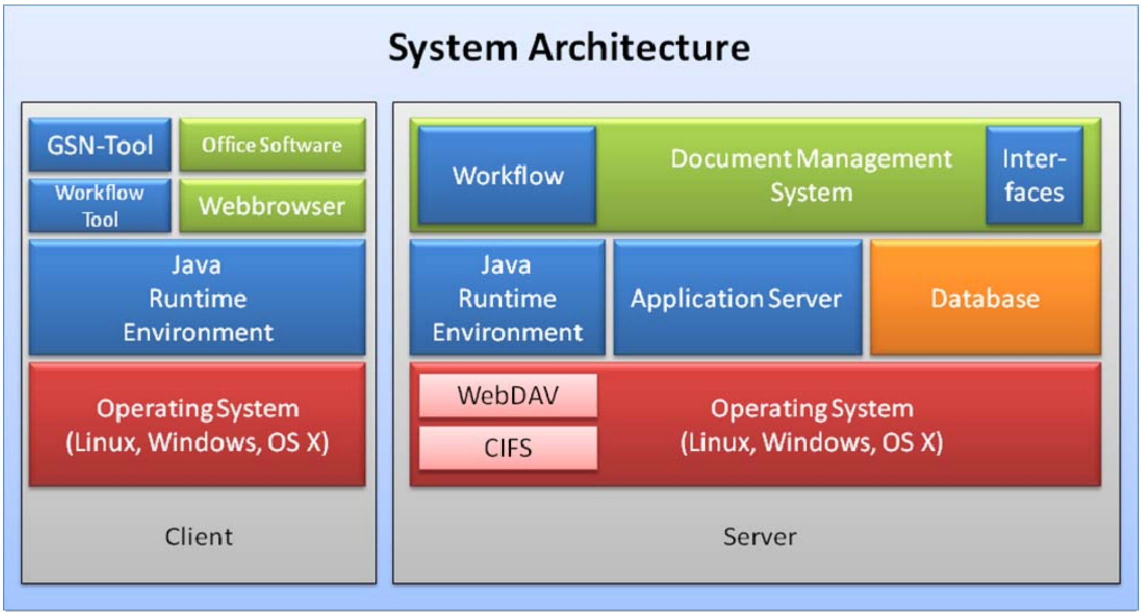


Figure (A): A possible System Architecture of the Tool to be developed in WS G

2.3.2 Programming and Configuration

To reduce the amount of programming, components should be chosen with the goal of interoperability (based on available standards). This means that a DMS should be able to operate on many different databases and, that workflows within the DMS should be configurable with existing standard tools.

The overall amount of programming (blue) and configuration (green) is shown in Figure (B). It is only necessary to

- setup users, groups and roles within the DMS
- define Workflows with an existing Workflow Tool for the DMS
- realize Interfaces (or a specification for interfaces) for the DMS to work together with existing Systems, or preferably use existing interfaces
- implement GSN-Tool

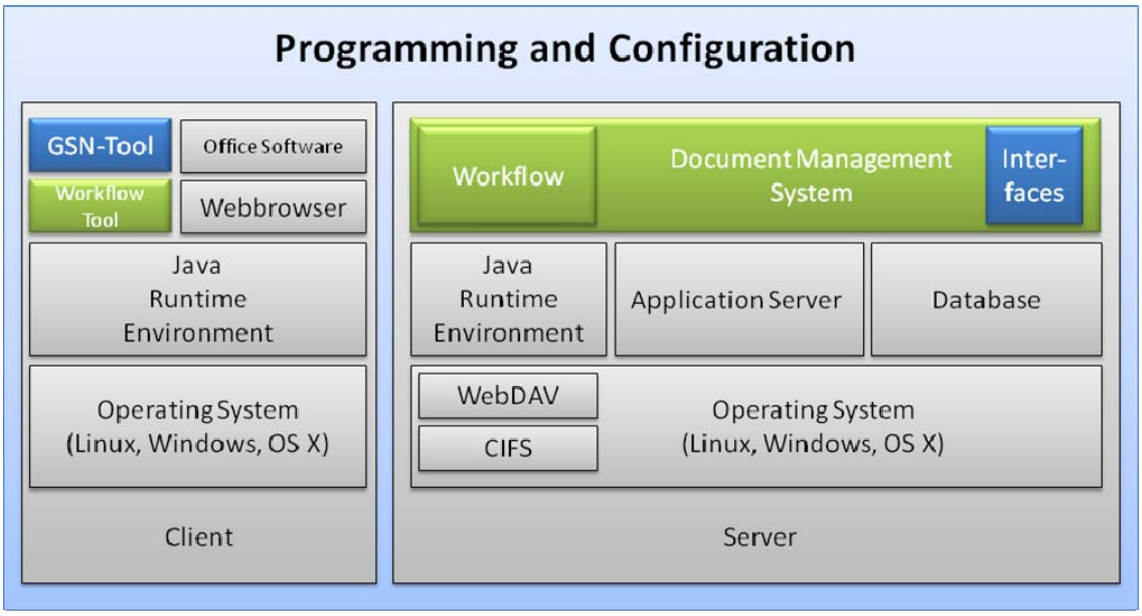


Figure (B): A possible system architecture of the tool to be developed in WS G

2.3.3 Workflow Tool

Major problems that were identified during the interviews and in the workshop were related to the workflow that the Safety Case Process imposes on a company's organisation. Even though a Safety Case Process implies a certain order of things to do and by whom they should be done, it is still necessary to adapt these processes to the company's specific needs and coordinated processes. This is usually done with Workflow Tools for BPM (Business Process Management). For example with such tools it is possible to model a "draft, review, approve/reject process" of documents related to the personnel involved. A user interface of such a tool can be seen in Figure (C).

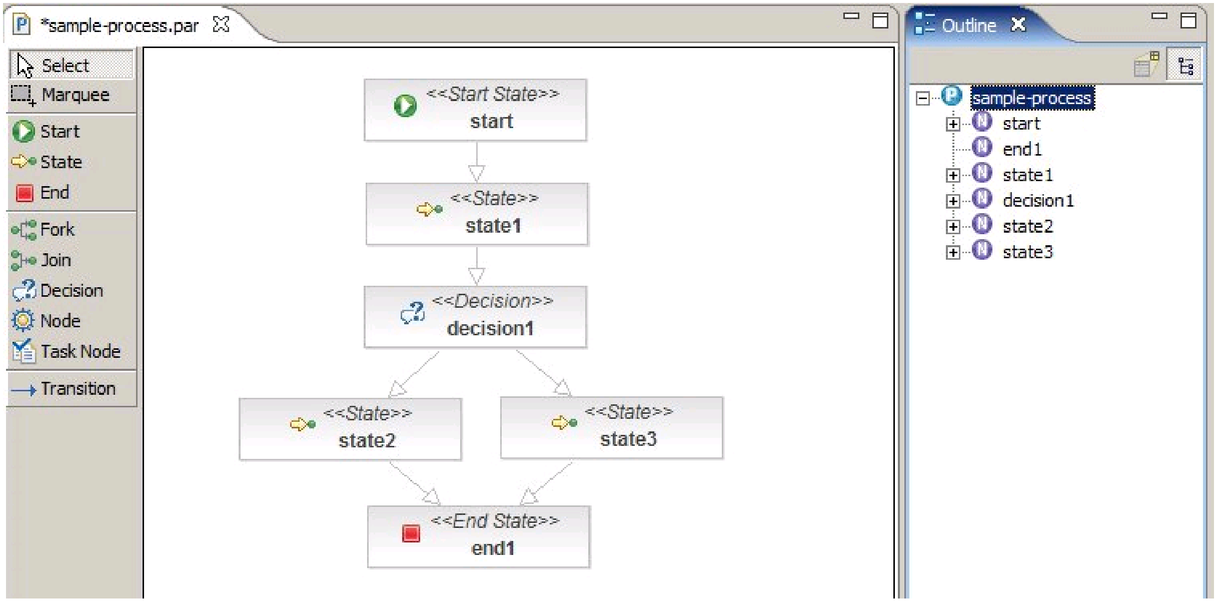


Figure (C): Screenshot of the workflow-tool

2.3.2 The use of the Goal Structuring Notation

2.3.2.1 The Goal Structuring Notation – an Introduction

The Goal Structuring Notation (GSN) is increasingly being used in safety-critical industries to improve the structure, rigor, and clarity of safety arguments. The purpose of a safety case can be defined in the following terms: *A safety case should communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context.*

The Goal Structuring Notation (GSN) (see e.g. [A], [C]) – a graphical argumentation notation – explicitly represents the individual elements of any safety argument (requirements, claims, evidence and context) and (perhaps more significantly) the relationships that exist between these elements (i.e. how individual requirements are supported by specific claims, how claims are supported by evidence and the assumed context that is defined for the argument).

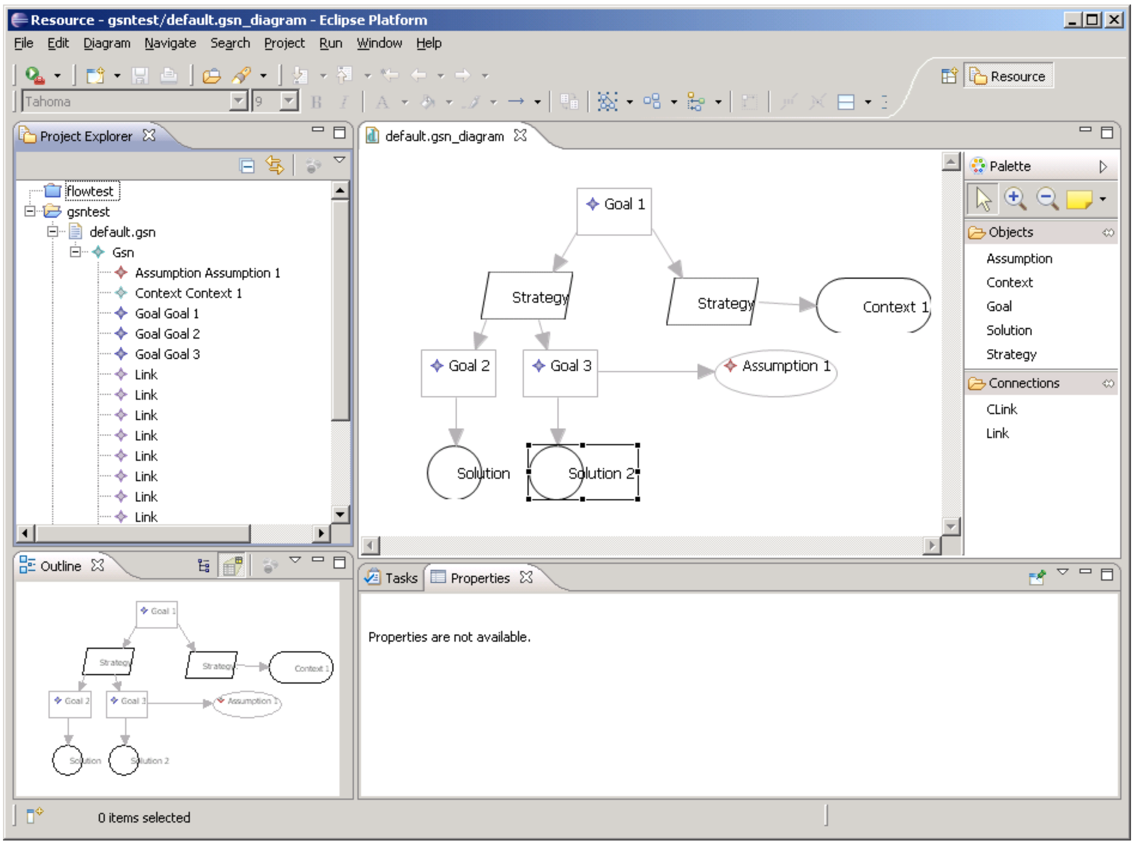
When the elements of the GSN are linked together in a network they are described as a '*goal structure*'. The principal purpose of any goal structure is to show how goals (claims about the system) are successively broken down into sub-goals until a point is reached where claims can be supported by direct reference to available evidence (solutions). As part of this decomposition, using the GSN it is also possible to make clear the argument strategies adopted (e.g. adopting a quantitative or qualitative approach), the rationale for the approach and the context in which goals are stated (e.g. the system scope or the assumed operational role).

The Goal-structuring Notation (GSN), developed by the University of York, provides a graphical means of setting out hierarchical Safety Arguments, with textural annotations and references to supporting Evidence.

The logical approach of GSN, if correctly applied, brings some rigor into the process of deriving Safety Arguments and provides the means for capturing essential explanatory material, including assumptions, context and justifications, within the argument framework.

2.3.2.2 A tool to define a Goal Structure

In addition to the DMS, a special tool (the tool to define a Goal-Structure – see also deliverable WS G, D 2.2) will be developed to support the safety argumentation. Therefore, such a tool provides a defined view on all the documents relevant for the Safety Case. This tool should be easy to use and build on things already learned or already necessary for the Safety Case Process. Therefore it was decided to use the same programming tools and methodologies to develop the GSN-tool, as for the already existing Workflow modelling tools. Furthermore the same programming tools are in use in other EU projects (e.g. AMPLE – Aspect-Oriented, Model-Driven Product Line Engineering) and relevant 'lessons learned' can be taken into account. As the programming tools had to be evaluated in a 'real life' environment, it was already possible to produce a prototype. A screenshot of the GSN-Tool prototype can be seen Figure (D).



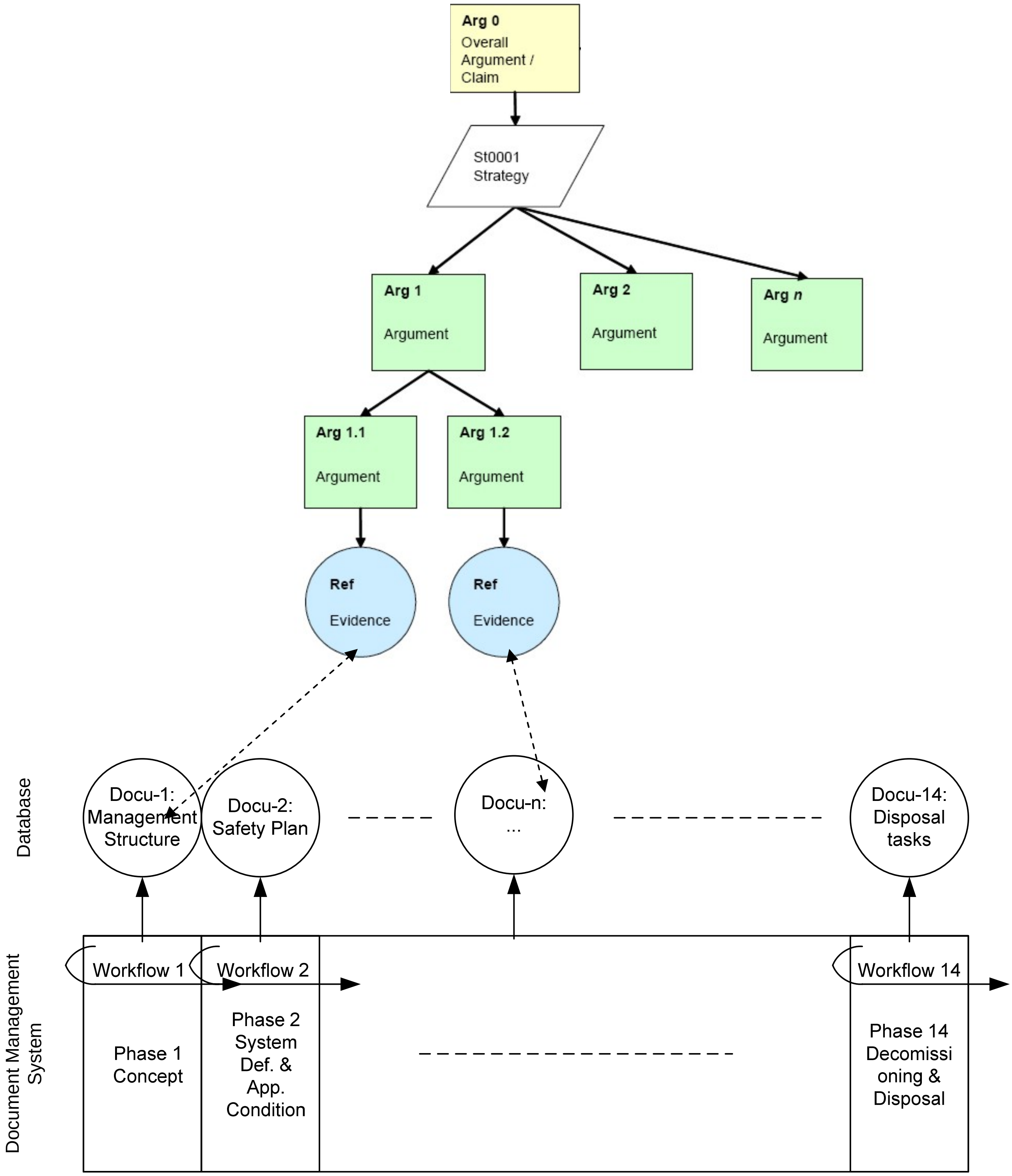
Properties are not available.

0 items selected

Figure (D): Screenshot of the GSN-tool

2.3.3 The Relation between the GSN Tool and DMS

The following figure depicts the general idea of the relation of the GSN Tool with the DMS. The Goal Structure will describe how the safety of a product has to be argued on the basis of the documents that are generated. Certainly, the Safety Manager has to guarantee the consistence between the structure of arguing and the content of the basic documents.



2.3.4 User Experience

A basic Idea of what the working structure of the tool will look like can be seen in the following figure. The DMS can be configured by an administrator. Workflows can be defined with a workflow-tool (by a user or by an administrator). Uploading of workflows would be done by an administrator.

The user can access the DMS and all its functions (versioning, tagging of documents, document search etc.) with all the known tools (Explorer, Office-Software, Web-Browser). With the new GSN-Tool it will be possible to configure a defined view on the documents to build a safety argument (see Figure (E)).

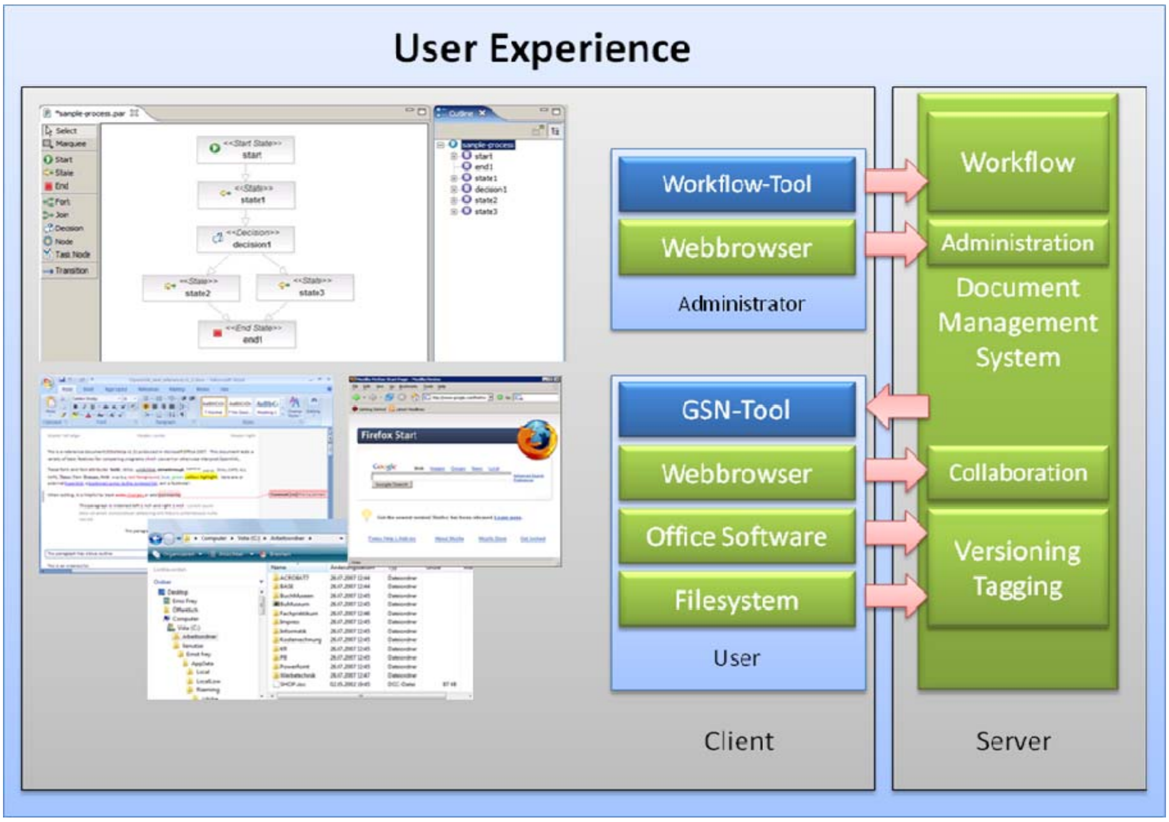


Figure (E): Screenshot of the GSN-tool

Section 3 – Conclusions

The presented overall system architecture was taken into account to evaluate open-source Document Management Systems. Once the decision for the DMS was clear, it could be shown, that many of the requirements identified during the interviews of Deliverable D.G.1.2 and during the workshops to prepare Deliverables D.G.2.1 (Long Term Goals) and D.G.2.2 (Short Term Goals) are already satisfied by the chosen system (e.g. document searching, document versioning).

The requirements described in section 2 of this Deliverable will provide the baseline for validation and verification in the tasks to come.

A modular approach based on open communication standards was chosen to provide a maximum of flexibility. The system will be independent of the type of operating system and components will be programmed in Java.

The requirements allow flexibility during the implementation phase but nevertheless clearly describe the wanted functionality of the software.

Drawings of a system architecture and prototype user interfaces are provided to give a clear vision of the end product.

Section 4 – Bibliography

[A] T. P. Kelly: Arguing Safety – A Systematic Approach to Safety Case Management, DPhil Thesis YCST99-05, Department of Computer Science, University of York, UK, 1998.

[B] Tim Kelly and Rob Weaver: The Goal Structuring Notation – A Safety Argument Notation, Department of Computer Science and Department of Management Studies, University of York, UK.

[C] European Organisation for the Safety of Air Navigaion: Safety Case Development Manual, European Air Traffic Management, Eurocontrol, 2006.

Section 5 – Annex

List of functions / Collection of requirements

The following list of functions is currently in a draft (i.e. commented) state, as other reviewers will have a look on them in the near future. Later on the list of functions will be split up and end up in the documents for long term and short term goals. For now they will stay here to document the current status.

5.1 Referred documents

Nr.	Doc- Name	Version / Date	Notice
1	INESS_WS G_2nd Workshop_TUBS	2009/06/16	
2	dkn_2009-06-02_Tool-functions-reqs_0-3_JS.doc	2009/07/01	

Headline, Nr, function and source are copied from /1/ if not noted otherwise. The description was filed with notes and comments from Robert Bass (BBR) and Holger Kaumann (BBR) to make the requirements more readable.

5.2 Internal information flow

Nr.	Function	Source	Description
1	Database of products and experts	Meeting at project start	That database is used to find a) the right team to handle a safety case b) a team of experts to handle exceptions of use and product faults.
2	BBS tool for meetings	Daily Meetings.	The bulletin board is a collector for various information. a) dates of milestones b) problems and their solutions c) a list of open points with dates and the name of the person responsible to complete this task
3	Calendar tools for Meetings	Quarterly Briefings, document review	[similar to 2]
4	Generate role and verification plan	Role and verification plan	The role plan includes a number of steps in the workflow and links them to a responsible person. a) Linking a person to one “step of the workflow” to show the responsibility b) Showing that the role plan is compliant to the EN50129. c) Showing that all included persons are experts for the current safety case.
5	Access to people based on roles links to them.	Role and verification plan	[similar to 4]
6	Simplify documentation, knowledge database	Knowledge in heads not documented	A lot of developers place their information in different Tools and documents to make this information available for everyone. A Tool could search in multiple sources. If there is no direct link to a solution, there could be e reference to a person to find help. This tool should search in hazard logs, Bug tracking- tools like Mantis, in Lists of open points (LOP) or even in documents. This kind of full text search will be slow but very helpful to search in older sources. There for a sort- function by date of source could be useful.
7	Database of products and lessons learned	Lessons learned: There is no structured method	[similar to 6]
8	Develop information flow structure	information flow unclear	To make the tool accessible in different companies the configuration of work and information flow has to be changeable.

9	Link to role and verification plan	information flow unclear	[similar to 4]
10	Automatic email notification of stakeholders	information flow unclear	If one step of the workflow is completed the responsible stakeholder has to inform the next. The automatic email notification should be designed to keep the workflow continuously running.

5.3 External information flow

Nr.	Function	Source	Description
11	Access to (relevant) Documents via inter / Intranet	Access to both Parties	To improve the communication between all people working on the safety case
12	Add acceptance criteria by country	list with major criteria for acceptance of a signaling systems	To make a product suitable for different countries, the acceptance can only be granted if the acceptance criteria is known.
13	Check certain criteria automatically	list with major criteria for acceptance of a signaling systems	The criteria could be found in EN 50129, EN50128. To check the criteria more easily the search function from Nr. 6 could be useful. The search function must be limited to the current project..
14	Choosing measures from EN50128	Separate document with measures from EN 50128	Selecting the measures and methods at an early state of the project, the assessor could be requested for acceptance. After the the assessor has accepted these measures and methods can be handled like a requirement.
15	Automatic skeleton document	Skeleton document of SaCa at beginning of project	<isn't it easier to place a write protected template of the mandatory folder collection filled with document templates somewhere in file system.?>

5.4 Existing Tools

Nr.	Function	Source	Description
16	Integrate Fracas functions	Fracas is used	The full name of Fracas is Failure Reporting Analysis and Corrective Action System ; this system could be used as a hazard log with additional functions for analyzing problems.
17	Get change information by mail	Doors automatic information about changes	Tools like Doors are able to create an email message if a former information has changed.
18	Simplify Requirements management functions	Doors complicated, oversized	Tracing requirements manually takes a lot of time and is error prone. Tools like Doors Keep all information in a database where the requirements are linked to sub requirements, architecture, tests and even test results.
19	Search function for all documents and versions	No search in document management system	[similar to 6]
20	Automatic versioning	Arbitrary version numbers in documents	After changing information in a SaCa the system should ask for a new version number. Therefore the author has to decide how to update the version number.
21	Tagging system for documents	Approved Versions are not marked	Some versioning tools are using a database to keep the meta information of a document together this could include version number, change notification, author and a label collection. For example this label collection is used to mark a collection of versions of different documents which are sent to the assessor.

5.5 General functions

Nr.	Function	Source	Description
22	Hints if modification of RAMS necessary	Keeping RAM-Documents up to date	[Similar to 17]
23	Online Help	Guide to using the standards considered to be helpful	/2/ A guide using the standards is considered to be helpful. → The tool could guide the user through the standards. (W)
24	Hints to examples, state what is needed	KISS: Keep it simple and save	/2/ KISS: Keep it simple and safe. Consider thoroughly what is really needed. → The tool could clearly state, what is needed. For some parts it might give good examples or recommendations. (W)
25	Links to other resources	SaCa should be a thin document	To make a SaCa- document easy to read and even to understand, it should include only the primer argumentation. It is easier to divide a SaCa in Sub-SaCas with an own documentation and closed SaCa argumentation. Referencing to Sub-SaCas reduces the volume of documentation and makes the safety case structure more clear.
26	Tool to predefine structure form EN 50129	Provide overall structure	To reduce documentation time the structure of documents and file or folder system shouldn't change form project to project. Keeping them similar makes generic documents for example the configuration or the quality management plan possible. In the same way a stakeholder knows after the first finished project where to place his information.
27	Unify SaCa for sub-Systems	SaCa structure and tests for subsystems	This is an extension to /25/ and /26/: If the SaCa is unified it is much easier to find the relevant information to reference to.
28	Hints what to do if SaCa missing	No SaCa for legacy systems	These hints are a collection of methods to use a legacy or any other black box system in a safety case. A selection of the methods could be handled like requirements of the current SaCa.
29	Overall function of tool should perfect SaCa	Role of independent safety assessor unclear	<-?->
30	Collect Costs of SaCa	No figure about Costs available	Most bigger companies are using a controlling system. Most systems are not configured, to collect the costs of a SaCa.
31	Link to lessons learned database	No error Database	[perhaps similar to 6]

5.6 Documentation

Nr.	Function	Source	Description
32	Tool to provide checklists	Checklist from verification plan	There are different checklist running in a safety case a) The list of open items. b) The formal checklist of workflow c) The checklist of measures … d) checklists for document verifications e) checklists for phase verifications only the checklists b), d) and e) can be provided by an tool.
33	Automatic collection of open items	Open items in separate section of document	Placing open items like a verification attachment at the end of a document will reduce the verification time. Sometimes the points of open items do not only belong to one document or the decision where to handle the points isn't made. In that Way open points could be lost or forgotten for a time.
34	Automatic versioning	Manual version numbers in documents	[similar to 20]
35	search function for all documents and versions	No search in documents management system	[similar to 6]
36	Interaction of databases	General Discussion	For a powerful search function multiple interfaces are needed. Each document type will crypt its information in a different way. This includes database too. Because if the interface is known (SQL perhaps) the structure of the knowledge is not known.
37	Show impact of changes in SaCa	General Discussion	Including 21: this function could be supported by the DMS. An impact change should be noticed in the meta information of the "head document" of a SaCa
38	Backup file automatically, enforce versioning system	Parallel files in versioning system and as Local Files.	[similar to 21]

5.7 Concept phase

Nr.	Function	Source	Description
39	Creation and update of risk analysis by tool	Risk analysis performed	<- ?-> How? Is this a hint for example to VDV331 where some risk analysis are collected. ?
40	Configurable workflows	General discussion	Possibility to configure the workflow makes the tool suitable for different companies. But in the same way different configurations can prevent easy data exchange.
41	Hint to open items on document close	General discussion	
42	Analysis of project environment	No structured analysis of project environment	If there is no analysis of the project environment, there are experiences from older projects. In that way each company has got an own skeleton of folder and file templates.

5.8 Requirement management

Nr.	Function	Source	Description
43	Tagging Requirements as safety relevant	Safety requirements are not Systematically identified	
44	Function?	Operators often do not provide a risk analysis	[similar to39]
45	Represent requirements structured / graphically	Determine completeness of requirements	
46	Automatic tracing of requirements	Manual tracing of requirements (with Excel)	
47	Provide Checklist for verification of requirements	No checklist for verification of requirements	
48	Archiving of technical documentation	Legacy systems often lack documentation	
49	Automatic collection of open items	No method to track open items or Argumentations	

5.9 Additions on the Workshop from 16.6.2009

Nr.	Function	Source	Description
50	GSN	University of York	GSN: Global Structured Notation is used, to argue with assessors and customers about safety this is useful for example in [25]. More information on: www-users.cs.**york**.ac.uk/tpk/dsn2004.pdf
51	Structure of hazard log	general discussion 090616	
52	Link to hazard log	general discussion 090616	[similar to 36]
53	Tracing of application conditions	general discussion 090616	For tracing the application conditions there is a continuous communication between railway company and supplier necessary. But this is hardly possible, because this takes time. Sometimes the supplier isn't requested because the time to decide is limited by train frequencies.
54	Configuration able of document states in the DMS	general discussion 090616	[similar to 21]
55	Interface to the remarks specification tools	general discussion 090616	[similar to 36]
56	Interface to the document management tool	general discussion 090616	[similar to 6]
57	Configureable skeleton document	general discussion 090616	[similar to 40 and 42]
58	Matrix of interested parties.	general discussion 090616	