Author(s): Nurminen, Miika; Suominen, Panu; Äyrämö, Sami; Kärkkäinen, Tommi

Title: Use cases for operational decision support system

Year: 2008

Version:

Please cite the original version:

Nurminen, M., Suominen, P., Äyrämö, S., & Kärkkäinen, T. (2008). Use cases for operational decision support system. In T. Mätäsniemi (Ed.), Operational decision making in the process industry - Multidisciplinary approach (pp. 107-131). VTT. VTT Research Notes, 2442. http://www.vtt.fi/inf/pdf/tiedotteet/2008/T2442.pdf

7. Use cases for operational decision support system

By Miika Nurminen, Panu Suominen, Sami Äyrämö and Tommi Kärkkäinen
University of Jyväskylä

7.1 Introduction

The task of decision making can be divided into three steps (Simon 1976): (1) the identification and listing of all the alternatives; (2) the determination of all the consequences resulting from each of the alternatives; and (3) the comparison of the accuracy and efficiency of each of these sets of consequences. Simon (1965) refers to the first of these as *intelligence* (in a "military" sense), the second as *design*, and the third as *choice.* Simon's division combines the organizational (descriptive, what decisions could and should be made) and technical (normative, how you should make the decision) views on decision making. The OODA Loop is another decision making model created by military strategist John Boyd (2007). The model is meant for organizations that undergo continuous interaction with their environment. The OODA loop consists of four overlapping and interacting processes, namely *observe*, *orient*, *decide,* and *act*, that are in continuous operation during the interaction.

During the consortium project we have shared a detailed documentation on the specification of generic operational decision support system (ODSS) which is based on statistical decision theory (SDT). This generic user requirements (GUR) document, as given in Appendix A, is based on ideas similar to those of Simon and further elaborated by (Jokinen et al. 2008), providing a comprehensive checklist for the development of any system supporting operative decision making based on SDT. However, as pointed out by Simon, organization-wide decision making is more than just a software realization of one decision support technique, so that an organization-wide DSS should be based on abstraction levels (layered architecture) separating decision task selection and actual decision making support in a modular way. From the enterprise architecture point of view (see e.g., Kilpeläinen 2007 and articles therein), a comprehensive description should initially focus on *contextual and conceptual* enterprise levels instead of physical or detailed representations in light of the classical Zachman's framework (Zachman 1987).

Moreover, the GUR description in Appendix A is conceptually rather "loaded" i.e. it contains a significant amount of different concepts with ambiguous meanings (e.g. referring to SDT elements). Hence, in this chapter, we augment the GUR specification with *business use case -like descriptions* (Cockburn 1997). Actually it is quite common (see

Bittner and Spence 2002) that functional specification of a system that is strongly based on one particular realization technique can yield a large descriptional bias.

We present the revised DSS specification in the form of use cases to support creation of a conceptual model. The use cases and the resulting conceptual model can be used to set fixed and common terms among the DSS stakeholders. The use cases, the conceptual model, and the reference models for decision support systems can be used to analyze the possible structure and abstraction layers of the general operational decision system being investigated. Further, the stereotypes and concepts discovered from use cases form a base for a domain-specific ontology that can be applied for information integration and automated reasoning about decision support systems.

The contents of this chapter are as follows: first, we provide an introduction to previous related research. Next, we present the use case specification for a generic ODSS. The use case specification is used to generate an entity model that describes the domain for decision support systems. Finally, the chapter is concluded.

7.2 Preliminaries

This section provides a short introduction to not yet covered related research from organizational, information systems, human decision making, and system specification perspectives.

7.2.1 The degree of digitalization and its impact on information systems

The amount, degree and form of communication used in organizations should be taken into account when designing decision support systems. With the current trends of digitalization and the convergence of networks, the amount of available information is higher than ever. Thus, defining and gathering necessary information is a crucial step in realizing decision support.

The digitalization trend has generated new problems and added to the impact of existing ones, such as information overload. The ease of information distribution, for example by overdistributing or forwarding mail to many people, can impair organizational communication by overloading the persons receiving the data with irrelevant or secondary information (Kilpeläinen 2007).

Despite the increased digitalization of documents, organizational information will never be available in its entirety for automated processing by decision support systems. In (Kilpeläinen 2007), organizational communication from three industrial and academic

organizations was analyzed. Overall, it seems that digital documents account for about 40–55% of total communication (depending of the measures used), leaving out analog representations (e.g. paper) and other communication (e.g. face-to-face, phone). Since some of the analog documents are produced digitally despite the medium used (e.g. printing documents), the actual amount of digital communication might be higher, but still a notable part of communication takes place outside the information systems.

Even if both digital documents and other communication forms are considered, tacit knowledge can not be directly accessed by a decision support system, even though it may have a pivotal role in decision making compared to official documentation. In principle, this can be alleviated by expressing tacit knowledge explicitly to become part of the organizational information resource, but in practice both measuring and acquiring tacit knowledge can be difficult and time-consuming. Therefore, one should note that any information system can have direct access only to a fraction of the total knowledge present in an organization.

7.2.2 On decision support systems

Decisions can be seen as a way of addressing a problem. All decisions contain some kind of procedure or chain of reasoning as to how the problem should be solved. If not, decision degenerates to merely guessing. However, the level to which the procedure can be automated or the so called structuredness of the problem can vary greatly. Basically one can define three categories based on the structure: structured, semistructured, and unstructured (Gorry and Scott-Morton 1971).

For structured problems there exists a known procedure (e.g. standard operating procedures and processes, operations research, electronic data processing and heuristics) to find the best or a good enough solution (Simon 1965). Semistructured problems have some parts that are procedurally solvable and others that are not. Unstructured decisions consist of seeking answers to problems that have no known and robust method for solving them. For example, planning for research and development is highly unstructured problem while locating a warehouse is a structured one.

To help solve these problems decision support systems can be employed. The DSS is designed to support the user in making a certain decision. Usually this is achieved through modelling a subset of the real environment and analysing possible outcomes of decision candidates. Sometimes just simple calculations are enough. DSS covers a broad range of applications from simple spreadsheets to sophisticated artificial intelligence systems – all having in common the goal to ease solving the problem they are designed to help with.

Turban et al. (2004) describe three essential subsystems of DSS (see Figure 7.1): data management, model management, and user interface (UI). Hence, DSS is constructed from data, ways to manipulate them, and an interface for the user to interact with the system. Additionally there might be a knowledge management system that provides intelligence for the system. As is often the case with general concepts, DSS subsystems are loosely defined.

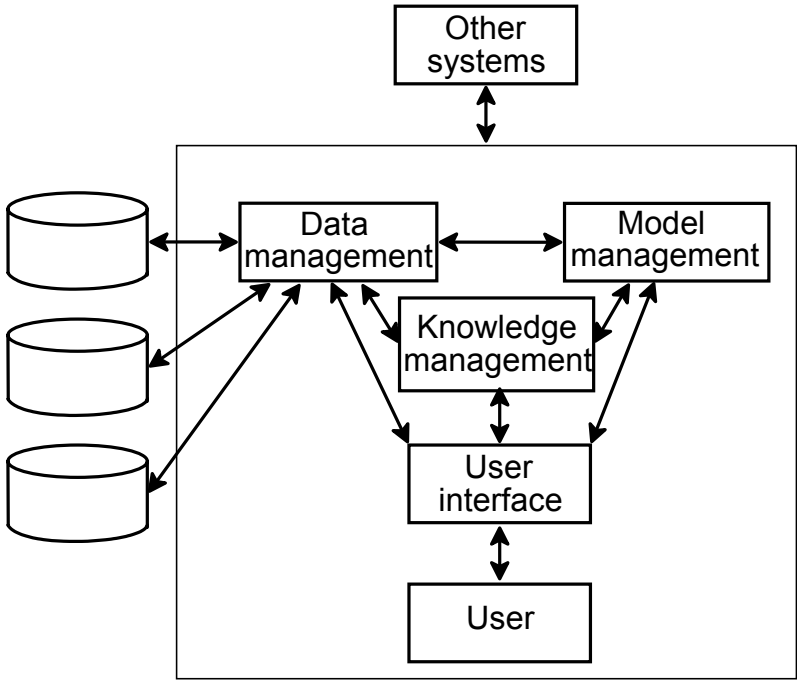

Figure 7.1. The general framework of decision support systems (Turban et al. 2004).

- **Data Management**. To make rational decisions some kind of (relevant) information is needed. Handling this data is done through a data management system. It is often based on some kind of database management system. In a corporate management environment the system could be connected to the data warehouse of the corporation to provide relevant information.

- **Model Management**. Models are routines that are made to provide some kind of analysis capability in DSS. They can be complicated simulations or just simple calculations that use information stored by the data management system. The model management system provides means to create, modify, and run the models. This requires the subsystem to be able to handle models similarly to data. Same database backend might be used also for model storing.

- **User Interface**. User interface enables handling models and information to support given decision task. Because of the close relation to human cognition user interface should be considered carefully to support the decision task and to minimize errors caused by misunderstanding data and analysis.

- **Knowledge Management**. Many problems require expertise to solve them and the results from an analysis can be difficult to interpret. In DSS this can be provided with a knowledge management subsystem. It is basically a collection of methods derived from artificial intelligence research that enable classification and heuristic evaluation of results or automatic problem solving.

7.2.3 Cognitive biases and decision making

Arnott (2006) claims that although influences of DSS on decision performance are often disappointing, focusing on decision-making and tailored support can lead to successful systems. Arnott perceives DSS to be fundamentally about decision making and thus a DSS analyst should have knowledge about human decision processes and how to improve them.

It seems likely that without knowledge of human behaviour the system will fail in helping to make the right decisions. For example, even if the system could give accurate answers for any given problem, people are not likely to follow them if they don't feel they are in control and understand the chain of reasoning behind the answers. This is because people are likely to overestimate their chance of success when they are in control (Mann 2002) even if they are not equipped for the given task.

The decisions made can vary from the most rational choice. Predictable deviations from rationality are called cognitive biases. Arnott classifies 37 biases into categories of memory, statistical, confidence, adjustment, presentation, and situation presented here briefly.

- **Memory biases** (*hindsight, imaginability, recall, search, similarity, testimony*) are mostly due to the fact that people remember and recall familiar events more easily than others. Such a human judgment then easily yields an incorrect estimation of possibilities. To help users cope with these tendencies DSS should provide information from the past and provide statistical information. User interface should take good care that figures are represented in a neutral way. Also every view should contain enough accurate information to deal with the current task, thus avoiding overloading users' short term memory.

- **Confidence biases** (*completeness, control, confirmation, desire, overconfidence, redundancy, selectivity, success, test*) arise mostly because of a decision maker's overconfidence in his/her skills. When underestimating the problem people tend to choose the first complete-appearing solution without considering alternatives. People are likely to look for confirming evidence while ignoring the search for disconfirming information. To address these problems DSS should show alternatives and present the uncertainty of information. Structuring should also reveal the difficulty of decisions. DSS should keep a record of the decisions made, enabling users to evaluate how successful they have been.

- People tend to be lazy and do not adjust enough to a change of environment. This kind of ignorance of potentially significant new data is categorised under **adjustment biases** (*anchoring and adjustment, conservatism, reference, regression*). DSS using up-to-date models based on recent data (on-line adaptation) provides the most reliable adjustment to the task at hand by the decision maker.

- The way information is represented can make a big difference. Scale differences between graphs can lead to wrong conclusions. First or last items in the list can be overweighted and so on. Problems arising from **presentation biases** (*framing, linear, mode, order, scale*) can be avoided by using consistent user interfaces with unified views.

- **Situation biases** (*attenuation, complexity, escalation, habit, inconsistency, rule*) include the human tendency to follow a previous unsatisfactory course of action, choosing an alternative only because it was used before. People are also eager to simplify the situation by ignoring or significantly discounting the level of uncertainty. DSS needs appropriate structuring (sequencing) of the decision tasks. Also history databases of decisions made earlier and their consequences should be stored and (re)utilized.

- **Statistical biases** (*base rate, chance, conjunction, correlation, disjunction, sample, subset*) result from misinterpretation of data that should be treated as random variables. Please refer to Chapter 2 for a discussion on compensation and avoidance.

Debiasing or compensating for the erroneous behaviour of the user should be considered when designing DSS, because these biases might alter the decision significantly. Alternatives to help the user overcome these shortcomings can vary from carefully considered user interface, statistical data, and representation of the probabilities, as well as just informing the user of common mistakes that people are likely to make in the current situation. If the problem can be structured this will help with these issues because the program is more able to follow the actions of the user. From a design point of view, methods for following users' behaviour should be implemented to track the success of debiasing strategies.

7.2.4 Use cases for system requirements

Use cases are a popular method used in the requirements elicitation phase of a software development process. Requirements elicitation involves acquiring information about SuD (System-under-Development). To get a complete picture of the requirements, they are considered with different stakeholders of the SuD. In requirements engineering lingo, a stakeholder is someone with an interest in the future system, e.g. a user, administrator, maintainer, etc. Use cases focus on describing the use of SuD as a part of workflows and business processes related to relevant stakeholders.

A use case is a description of the desired functionality of SuD in a given situation. According to Cockburn (2000), "A use case captures a contract between the stakeholders of a system about its behavior". A use case provides an important context for the distinct functional requirements, how they are connected, the situations they are relevant in, and the related trigger conditions. The level of detail of a use case varies widely and can be adjusted on a per-project basis. Also, the details are usually added in breadth-first, starting with the names of all the use cases and proceeding as far into detail as needed, usually by assigning attributes such as priority, success guarantees, etc.

Use cases do not describe the so-called non-functional requirements of SuD. These include measurable conditions and constraints related to e.g. performance, security, data requirements (Lauesen 2002), user interfaces, etc. Thus, use cases are not sufficient means to document all the requirements of a software system. Also, use cases are not well suited for all systems, e.g. reactive systems which constantly observe the surrounding environment and act accordingly (embedded real-time systems) (Jackson 2001).

Use cases have structurally much in common with business processes and workflow specifications, although the semantics, detail, and scope differ. Use cases usually focus on the interactions between the user and the system, whereas workflows and business process models tend to describe more general, higher-level activities – often omitting detail in the process models. For example, Sharp and Dermott (2001) utilize use cases to elicit system requirements for specific steps in a process model. However, since a business process can be defined as a specific ordering of work activities across time and place with a beginning and an end containing clearly defined inputs and outputs (Davenport 1993), at a syntactic level both use cases and business processes can be modelled with a graph structure. Furthermore, Cockburn points out that any system that offers a set of services for outside actors while protecting the interests of the other stakeholders can be described with use cases. This includes business systems.

Writing style, conventions, and consistent terminology are essential when considering the understandability of the use cases and the effectiveness of automated postprocessing of the models. Postprocessing techniques include data mining (Nurminen et al. 2005) and natural language processing (Kärkkäinen et al. 2008). This can be a challenging task in itself, because different people tend to produce different models even given the same domain (Soffer & Hadar 2003). To alleviate this, use cases presented in the next section were written in an iterative way with multiple reviews.

7.3 DSS specification

Next, a use case based specification of a generic (hypothetical) operational decision support system is presented. The GURs in Appendix A were the starting point of the specification but we present revised use cases with hierarchical layers and somewhat simplified writing conventions to ease the understanding of key functionality. Arnott's biases (2006) are also accounted for in the specification. The purpose of the revised use cases is to provide easily understandable material for communicating about System under Development without loss of accuracy.

The use cases presented establish a connection between the organizational level decisions of which tasks are to be handled by DSS, and the actual decision making with DSS. In this way, the two main foundations of operational decision making, i.e. technical support for decision making in the form of a formal decision making model (normative decision making) and the organizational thinking (descriptive decision making) point of view are both captured. To this end, the specification of DSS is the main concern here, but to have such a system in active use as part of everyday organizational operations requires resources and processes for systems' maintenance – the part (i.e. the lack) of the software/information system lifecycle which often causes the bad user experience.

7.3.1 Use case model

Our manual inspection of generic user requirements leads to the basic structure for SuD presented in Figure 7.2. The process starts with the need for the decision. This need can be triggered automatically by the system or specified manually by the user. If user sees the need for the decision there might not be a proper model and the decision task has to be modeled before any further action. After the model exists the DSS is able to generate a decision proposal. In some cases this is not possible and the user is able to work with the data, models, and structures that are available without the proposal (semistructured decision problem) (Gorry & Scott-Morton 1971). In the decision making process the decision maker is able to change which data are used, and possibly the parameters in the model to support the evaluation of the alternatives.

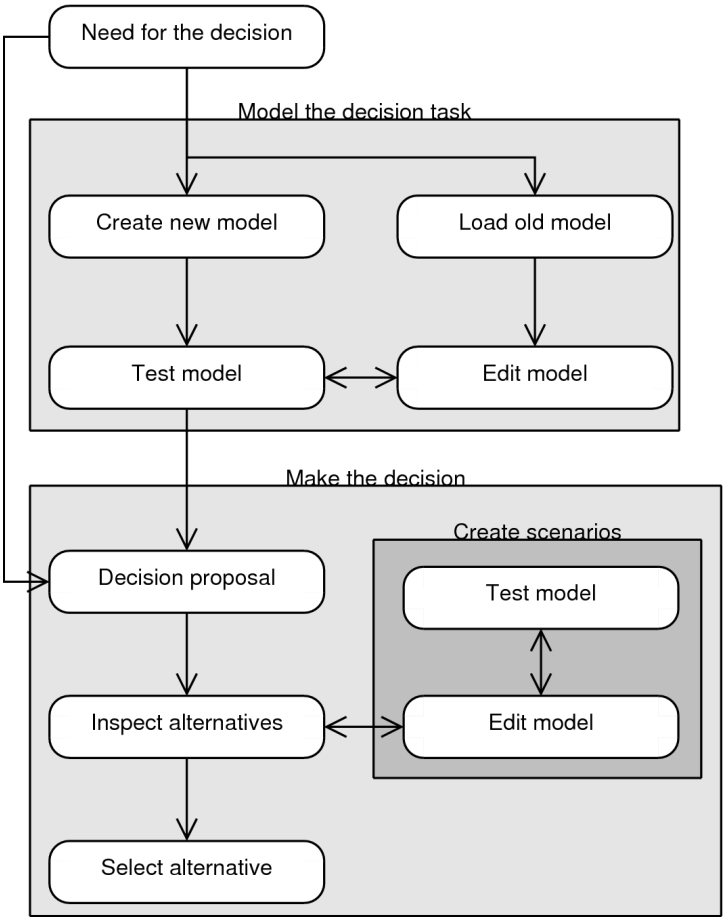

Figure 7.2. Basic structure and expected workflow for a decision support system.

The program main flow scenario in Figure 7.2 does not take into account the group decisions and some configuration steps that precede the usage of DSS. For example, defining data available to DSS is a requirement that is missing from this flow representation (GUR-2.1.2b in Appendix A). The steps for a single decision maker are considered in use cases and group working can be represented as variations (e.g. differing in actors or certain steps) to original scenarios. While these variations are important when a real system is developed, they make little difference when building a conceptual model because the variations by definition share many of the main steps of the use case.

The use cases presented here are generic. The actual configuration and location of systems and databases depend on the target organization. For example, if a workflow system with well-defined process descriptions is already present, it may provide detailed information about potential decision tasks to DSS. In addition, the evaluation of consequences depends essentially on the nature of a decision task. Some decisions have an instant, measurable outcome that can be detected automatically (e.g. on-line quality measurements of the end product in a product line), but most decisions are of a more abstract nature (e.g. financial, strategic decisions) that cannot be determined or even executed in a short time period.

Roles

The following generic roles are applied in the use case descriptions. Depending on the organization and the position of a decision maker, it is possible that more than one role is performed by a single person.

- **System Expert** is responsible for the DSS maintenance and configuration, data connections, software/method extensions and updates. This role presumes extensive technical skills in information technology, software engineering, and to some extent, in knowledge discovery, data mining/analysis, and statistics. The tasks of the System Expert might be partly or fully outsourced.

- **Decision Configurator** is responsible for the availability and storage of necessary data sources. He/she analyzes the information sources/flows in the organization and responds to the data requests from the Method Expert and/or the Decision Maker. The role presumes extensive skills in information technology, data engineering, and knowledge management. To some extent, skills in software engineering might be also needed.

- **Method Expert (Analyst)** is responsible for applying the computational and statistical methods of DSS to the target datasets. He/she has extensive knowledge in selection, usage and configuration of the methods. This role presumes extensive skills and deep understanding in optimization, simulation, data mining/analysis, statistics (incl. SDT), and other related methods. The Method Expert must be able to communicate about technical issues with the System Expert/Decision Configurator and, moreover, with the Decision Maker about the meaning of the results and representations. The tasks of Method Expert might be partly or fully outsourced.

- **Decision Maker** is an experienced domain specialist who makes decisions and is usually responsible for the outcomes. The Decision Maker is not expected to have detailed technical-level understanding of decision support methods or models, but based on domain experience he/she can evaluate the impact of different decision alternatives provided by domain and system experts, or the DSS.

- **DSS Configuration Team** combines the appropriate level of management and selected experts representing the aforementioned roles. The team maintains the decision support system by analyzing the need for supporting new decision tasks, decision making principles, methods, models, and pre-configuring decision tasks templates that guide predefined decision making tasks.

Information systems

DSS contains or interfaces with following libraries, databases, and other information systems:

- **Method Library** contains decision support techniques, such as large-scale data mining/analysis (clustering, neural networks, association rules etc.), SDT, (multiobjective) optimization, dimension reduction methods, and visual representation techniques. New methods can be added to the library by the System Expert. Utilization of the methods requires the tuning of parameters (distribution parameters for state estimation model, prototypes for clustering model etc.) or retrieving them from the Decision History Database, and the testing (validity, sensitivity etc.) of parameters. The Method Library is roughly equivalent to the Knowledge Management Subsystem in Turban's framework.

- **Decision History Database** contains data about previous decision support processes and analysis steps that are supported by the system. These data include the relevant information about decision making cases (date of problem, problem description, short-/long-term consequences etc.), analyzed data sources, operational tasks and method selections, input parameters (optional) of the applied methods, and obtained models (alternatives) with parameters. This database enables repetition of the previous decision support cases for new data and parameters. The consequences of the accomplished actions must be gathered for reusing the cases. Non-direct consequences are reported to the database later by the Decision Configurator or the Decision Maker. Direct outcomes are collected into the database automatically if possible.

- **Decision Template Database** consists of predefined decision making tasks that can be used to guide the decision support process. Each template defines the method selections, appropriate parameter settings, perhaps pre-adjusted models, visual representations, and informative descriptions. The templates are defined by the DSS configuration team. The templates are entered into the database by the Decision Configurator. The Decision Template Database is equivalent to Turban's Model Management Subsystem.

- **Organizational Data Sources** are information systems and databases that are used in the day-to-day operation of the enterprise. These provide the input data for the decision support system for analysis. The System Expert is responsible for providing connections to data sources. If the data from Organizational Data Sources is gathered to a permanent data warehouse to be used by DSS, this would be equivalent to Turban's Data Management Subsystem. However, ad-hoc usage without a dedicated data warehouse should also be possible, depending on the analysis methods used.

Conceptual stereotypes

The concept glossary (i.e. informal definitions for concepts) is needed to establish a joint language between stakeholders (managers, developers, users etc. related to DSS) (Bittner & Spence 2002). We base the glossary on use cases thus providing not only documentation about the existence of a concept but also its context of use. Moreover, attaching a stereotype to each concept creates a classification of them, supporting the critical transfer from domain analysis into system development. The introduction of stereotypes also clarifies the structuring of use case flow, because joint concepts related to system usage and its realization are tagged (Cockburn 2000). Moreover, there is no need to prolong the use case main scenario by repeating the user action and system response in connection with the same concepts (Wirfs-Brock 1993). We recommend that for a shared information transfer step between user and system ("Actor creates X" → "System stores X") the *use* case should be described from user's (usage) perspective only ("Actor creates X", tag X as persistent data stored by the system).

Table 7.1 contains definitions of the stereotypes that were used to classify the concepts. DecisionModelElement is specific to Decision Support Systems domain; other stereotypes are domain-independent.

Table 7.1. Definitions of the stereotypes that were used to classify the concepts.

Stereotype	Description
Action	Functionality needed by SuD
Data	Persistent information used internally by SuD
Database	Database to be managed by SuD
Document	Document to be produced by SuD or a report that SuD must generate to a user
ExternalAction	An external action that SuD must take into account
ExternalData	Data stored by other systems available and necessary for SuD
ExternalRole	External human or device that SuD must communicate with
Metadata	Data about data
Process	A specific ordering of work activities across time and place with a beginning and an end containing inputs and outputs
Role	Stakeholder role (the classification of a set of stakeholder representatives who share the same roles and responsibilities with respect to the project)
Selection	A particular choice related to a particular UserElement
System	SuD or other information system related to use case
UserElement	An element representing the interaction interface between a user role and SuD
DecisionModelElement	General entity related to the decision making model

Use cases

The overall structure and primary actors of the use cases are presented in Figure 7.3. Use case 1, *Perform Organizational Configuration and Decision Making Processes* presents the general process of utilizing decision support system in an organization and includes other use cases that are expected to be performed in an iterative way: Decision Tasks must be modeled before Decision Makers can use the system to make decisions. Finally, the Decision Maker can propose configuration change requests that can be implemented by the Configuration Team in the ongoing process of maintaining DSS.

Use case steps and related concepts are presented in Tables 7.2–7.9. Each use case may include notes that provide details to individual use case steps and references to related chapters. Numbers in parentheses denote links to another use case.

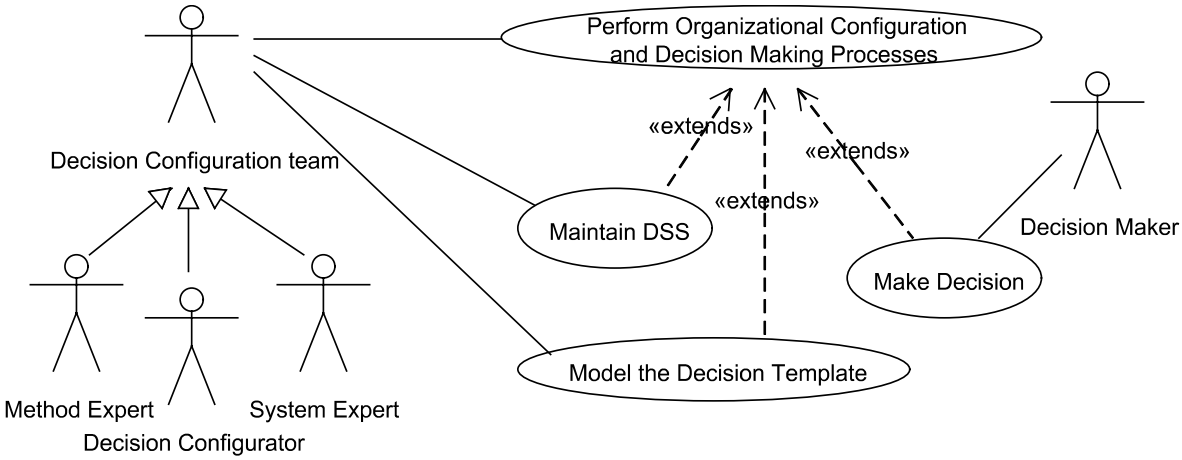

Figure 7.3. Use cases for DSS.

Table 7.2. Use case 1 – Perform Organizational Configuration and Decision Making Processes.

Id	Description	Concepts: Stereotype
1	DSS Configuration Team defines the set of organizational Decision Tasks to be supported by DSS and maintained by Decision Template Database.	DSS Configuration Team: Role Decision Task: UserElement DSS: System Decision Template Database: Database
2	DSS Configuration Team defines Necessary and Available Information for Decision Tasks.	Necessary Information: Data Available Information: ExternalData
3	DSS Configuration Team defines the set of available Decision Support Techniques to Method Library.	Decision Support Technique: DecisionModelElement Method Library: System
4	Method Expert documents the Decision Support Techniques to Method Library.	Method Expert: Role
5	DSS Configuration Team defines content of Decision History Database.	Decision History Database: Database
6	Decision Configurator Team models the Decision Templates (2) which are supported.	Decision Template: DecisionModelElement
7	Decision Maker makes Decisions (3) supported by DSS.	Decision Maker: Role Decision: Action
8	DSS Configuration Team maintains DSS (4) based on Configuration Change Requests.	Configuration Change Request: Document

Table 7.3. Notes for use case 1.

Step	Note
1	Selection of Decision Tasks to be supported by DSS can be based on e.g. critical task analysis (see Chapter 5), available data, existing knowledge sharing technology (e.g. digital diary between shifts) or process simulation models, the criticality of a decision concerning operative actions, the decision maker's capabilities and motivation etc. (cf. organizational thinking in Chapter 3).
2	Available Information refers to relevant (secondary) digital information, e.g. measurement data, laboratory analysis results, performance summaries, O&M reports etc. stored by existing systems. For limitations, see Section 7.2.1.
3	Possible techniques are described in Chapters 2 and 3 and references therein.
4	Some expertise is needed e.g. to introduce SDT portfolio for organizational unit management, as proposed in Chapter 2.
5	Here meta-information and comments can be attached to a decision to be stored along with reference to the applied decision support model.

Table 7.4. Use case 2 – Model the Decision Template.

id	Description	Concepts: Stereotype
1	DSS Configuration Team derives a generic Decision Task from past decision support cases.	DSS Configuration Team: Role Decision Task: UserElement
2	Decision Configurator checks the availability of relevant internal/external task-specific data.	Decision Configurator: Role
3	Method Expert attaches the Decision Support Technique suitable for the Decision Task to the Decision Model and notifies about necessary but missing connections from DSS to Organizational Data Sources in DSS.	Method Expert: Role Decision Support Technique: DecisionModelElement Decision Model: UserElement Organizational Data Source: Database DSS: System
4	System Expert creates the necessary but missing connections to Organizational Data Sources.	System Expert: Role
5	Decision Configurator specifies Trigger Condition for recognizing the need to perform the Decision Task.	Trigger Condition: Action
6	Method Expert defines the suggestive Decision Model Parameters for model building (distribution models, visual representations etc.) and inputs the parameters into the Method Library.	Decision Model Parameter: DecisionModelElement Method Library: System
7	Decision Configurator describes Decision Objectives and Decision Alternatives.	Decision Objective: DecisionModelElement Decision Alternative: DecisionModelElement
8	Decision Configurator attaches a structural Decision Making Process (i.e. phases or stages) yielding to a Decision Proposal for each Decision Task and stores it in the Decision Template Database.	Decision Making Process: Process Decision Proposal: UserElement Decision Template Database: Database
9	System Expert runs test cases (e.g., using earlier decision support cases) and reports the results to the Method Expert.	
10	Decision Configurator documents the elements of the Decision Model and its relation to Decision Support Technique in Concept Documentation and stores the Decision Model, its Concept Documentation, its testing and version history in the Decision Template Database.	Concept Documentation: Document

Table 7.5. Notes for use case 2.

Step	Note
	Steps 5–8 can occur many times in any order.
3	Method Expert might decide to load an existing model to be the base of the model creation. Decision Model includes relevant data for the Decision Task to be used with the Decision Support Technique.
5	Triggers for performing Decision Tasks are elaborated in Chapter 2.
6	In the case of data clustering (described in Section 3.1) as a decision support technique, this step means the estimation of clusters and prototypes comprising the decision model with chosen data. Uncertainty of the obtained clusters (state estimates) and consequent actions can be evaluated using methods of the statistical decision theory (Chapter 2).
7	This can mean the attachment of different control parameters to the current and desired state and consequent state alternatives and their probabilities.
8	The process can be sequential or parallel, relying on a single decision maker or a group of experts. Subtasks related to SDT process are described in Chapter 2. In case of clustering, prototypes are here interpreted (classified) according to KM process in Section 3.1, and the proposed decision alternative is attached to each of them.

Table 7.6. Use case 3 – Make Decision.

id	Description	Concepts: Stereotype
1	DSS detects a Trigger Condition for a need for decision and shows Decision Proposal to Decision Maker.	DSS: System Trigger Condition: Action Decision Proposal: UserElement Decision Maker: Role
2	Decision Maker selects Decision Alternative to be inspected.	Decision Alternative: DecisionModelElement
3	DSS shows Decision Model information related to the Decision Alternative.	Decision Model: UserElement
4	Decision Maker inspects and alters the Decision Scenario related to Decision Alternative. Decision Maker can propose a Configuration Change Request.	Decision Scenario: UserElement Configuration Change Request: Document
5	DSS generates and shows new Decision Alternative.	
6	Decision Maker makes Decision, documents it, and stores the Session with its Decision Documentation to Decision History Database.	Decision: Action Decision Making Process: Process Decision Session: Data Decision Documentation: Document Decision History Database: Database
7	DSS captures all relevant Consequences of the Decision made, if possible.	Consequence: Document

Table 7.7. Notes for use case 3.

Step	Note
	Steps 2–5 can occur in any order and many times.
2	Alternatives can be, for example, state change history of the cluster model that documents the influence of the different actions (process control adjustments) with respect to states (clusters).
3	This information can be an illustration of posterior probability densities for SDT (Chapter 2) or a visualization of process data and cluster evolution (Section 3.1). For example, taking action A when the process is in cluster (state) 1 leads to the state change from cluster 1 to cluster 3 with 90% probability and to cluster 5 with 5% probability. Decision Maker can also explore the previous decision making sessions, their decisions and the resulting consequences. These are recommended in the order of relevance related to current Decision Alternative.
4	When using clustering, the Decision Maker could change or request a change on the number of cluster prototypes (state estimates), clustering principle (e.g., different distributional assumptions) etc. (Section 3.1).
6	Decision Maker may accept a Decision Alternative, decide not to make a Decision, or cancel the Decision Making Process.

Table 7.8. Use case 4 – Maintain DSS.

Id	Description	Concepts: Stereotype
1	DSS Configuration Team receives Configuration Change Request related to the set of supported Decision Tasks from Change Requester.	DSS Configuration Team: Role Change Requester: Role Configuration Change Request: Document Decision Task: UserElement
2	DSS Configuration Team accepts or rejects the Configuration Change Request based on stored Decisions in Decision History Database and available information on documented Consequences of Decisions made.	Decision History Database: Database Decision: Action Consequence: Document
3	Method Expert modifies the set of available Decision Support Techniques and stores the results to Method Library.	Method Expert: Role Decision Support Technique: DecisionModelElement Method Library: System
4	DSS Configuration Team modifies the set of organizational Decision Tasks.	
5	DSS Configuration Team documents the changes in Decision Tasks, stores the Decision Tasks and Change Documentation to Decision Template Database, and notifies the Change Requester and other relevant users.	Change Documentation: Document Decision Template Database: Database

Table 7.9. Notes for use case 4.

Step	Note
1	DSS Configuration Team should have regular meetings to assess DSS and change requests.
2	If possible, DSS captures the Consequences of the Decisions. Consequences can also be documented manually.
	Steps 3–5 are performed only if Configuration Change Request was accepted in Step 2.
3	New, but presumably more complex computational tools and techniques appear rapidly and regularly.
4	This is an example of learning organization.
5	DSS Version Control database itself creates organizational memory concerning DSS life-cycle. Learning from the past can be supported e.g. by text mining techniques, e.g. (Nurminen et al. 2005).

7.3.2 Entity model

Use cases describe the problem domain in one viewpoint. The information is mostly not properly organised for software development. The development process can be further facilitated by extracting a domain model from the use cases. We encoded the use cases in ProcML – a semistructured XML format that allows attaching metadata to use case steps, such as conceptual stereotypes *role* and *database* (Nurminen et al. 2007). It is also possible to transform the specification to a website, allowing easy searching and browsing of the use cases. Use cases expressed in XML were subsequently analyzed by UCOT (Use Cases to Original entities) software (Kärkkäinen et al. 2008) to automatically generate a conceptual model based on the analysis. A grammatical parser (http://nlp.stanford.edu/software/lex-parser.shtml) and Abbott's heuristic (Abbott 1983) were used to process the use cases. In this section, we describe the entity model and evaluate the modeling process.

Figure 7.4 illustrates an unmodified, automatically generated entity model. As such, the model is not very useful because of the limitations in heuristic and natural language parsing. After initial processing the conceptual model was refined manually using UCOT by merging duplicate entities and dividing entities that represent multiple concepts. Subsequently, attribute and relation information was adjusted to reflect the actual application domain. A few nonessential entities and relations were omitted to make the model easier to understand. Finally, stereotypes were added to some of the concepts. The final model is illustrated in Figure 7. and shows approximately how different entities of the system act together. The model can be used in subsequent development phases of the system.

Although the use cases were mostly written using strict conventions (e.g. using subject-predicate-object structure), it proved to be exceedingly difficult to stick with simple sentence structures. The use cases were iterated many times with four different authors and as the domain understanding increased, the complexity of the sentences increased as well. For example, clauses like "if necessary" were added and multiple related actions of a single actor were combined to a single step. A specific problem (that can still be seen from the final model) was the complex relationship between Decision Support Technique, Decision Model, and Decision Task. They are referred to in many use case steps and often in an ambiguous way (e.g. "Method Expert attaches the Decision Support Technique suitable for the Decision Task to Decision Model") that is difficult to interpret automatically.

A known limitation in UCOT data model is the lack of support for n-ary relations. Since the use cases contained many instances of 3-ary relations (e.g. "Decision Configurator stores Documentation to Decision Template Database"), we had to divide the relation to multiple elementary relations (e.g. "Decision Configurator stores Documentation" and "Documentation is stored to Decision Template Database"). In addition, the variation of singular and plural forms, as well as the use of pronouns ("Decision Maker makes Decision and documents it") yielded unnecessary entities that had to be merged. Overall, the system was not very effective in processing long sentences and produced entities that actually contained either multiple concepts (e.g. "Decision Objectives and Decision Alternatives") or both a concept and a relation (e.g. "DSS based on configuration change requests").

Although it is relatively straightforward to "clean up" the model with UCOT after initial processing, maintenance becomes an issue if the use cases are modified after the entity model is modified manually. Since the relations from the entity model are not explicitly linked back to the use cases, it may be necessary to recreate the entity model from scratch after modifications are made in the original use cases. UCOT records all user actions after the model is loaded, so in principle it could be possible to apply some of the changes to the entity model automatically. Another possibility is to extend the ProcML data model with full entity linkage: as the use cases are processed, UCOT would tag each word with related entities. If the use cases are modified, the entity data would be preserved in XML descriptions. Both approaches should be considered for future development.

Based on the entity model, it seems that the roles "DSS Configuration Team", "Decision Configurator", and "Method Expert", as well as databases "Decision Template Database" and "Decision History Database" are highly connected. Other key entities include "DSS", "Documentation", "Decision Task", "Decision Support Technique", and "Decision Model". As noted earlier, many entities starting with word "Decision" are probably more connected than they actually need to be, so careful analysis of their actual relations is needed in the

subsequent development phases. At the current state, the model is not as understandable as we had hoped prior to use case specification. However, some hints about the required architecture can be discovered. For example, the activities of "Method Expert" and "System Expert" related to "Decision Support Technique" and "Organizational Data Sources" are somewhat isolated from the rest of the system, so they are candidate entities to be supported as separate (possibly outsourced) components. On the other hand, because of the high connectivity of "Documentation" to many roles and other entities, it might make sense to construct a common documentation system or format to be shared by different roles and subsystems – to be eventually stored in the Decision Template Database.

We emphasize that the modified model is by no means "final" – it merely provides a base for further development phases and should be updated as requirements or use cases change. Being generated from informal descriptions, the entity model does not necessarily represent the exact entities and relations (cf. ER-diagram used in database development) in the system, but helps to find the most essential entities (e.g. entities that are densely connected) that should be concentrated on. Depending on the development methodology, the model can be utilized in various ways. Perhaps the most common way would be to proceed with object-oriented analysis and design, separating classes and objects from the entity model and extending it with more technical detail. The entity model could also be generalized to a domain (meta)model to represent a set of requirements that are common for a set of applications, thus helping the creation of a software product line or a domain-specific ontology.

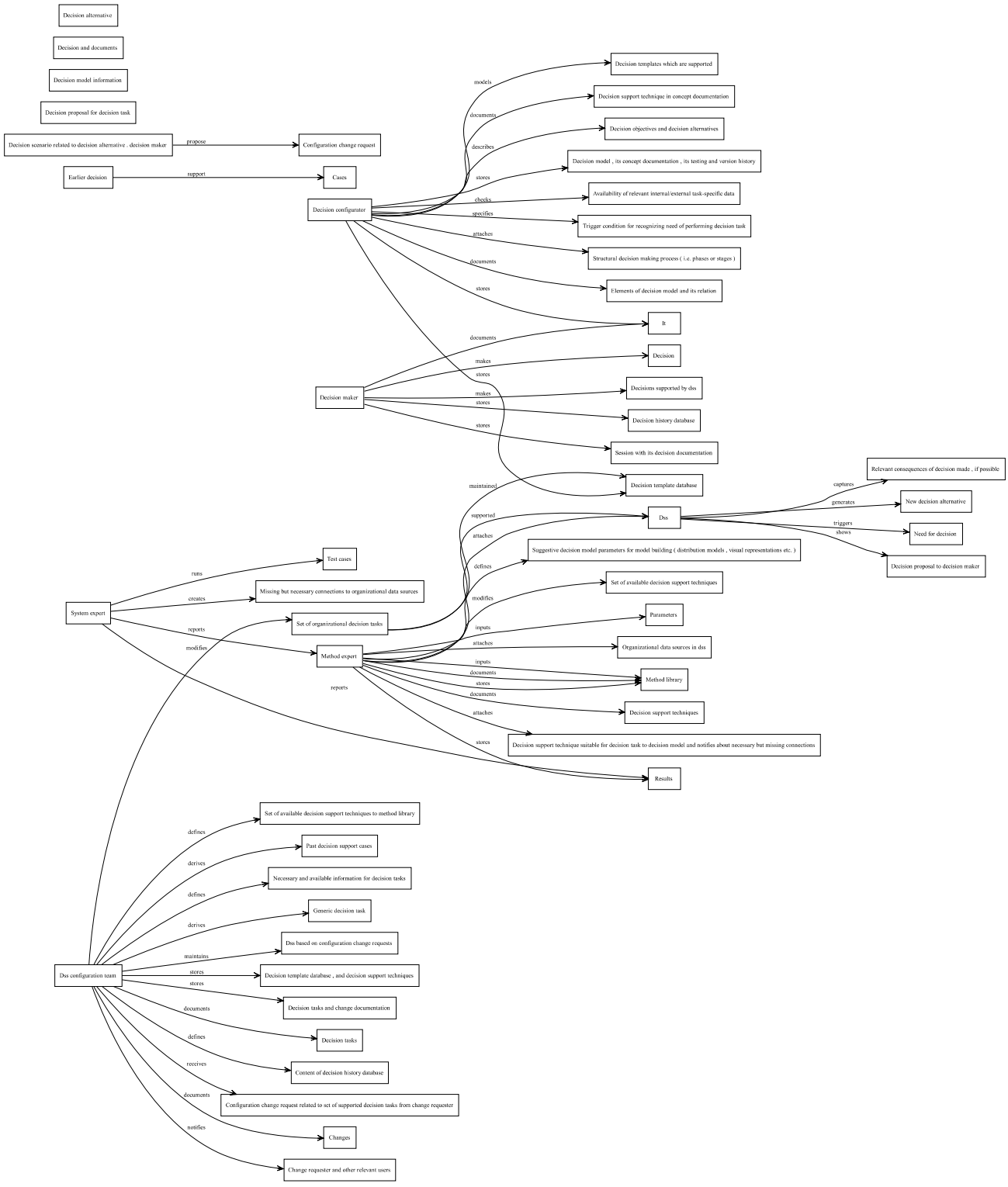

Figure 7.4. DSS entity model based on use cases – initial, automatically generated model.

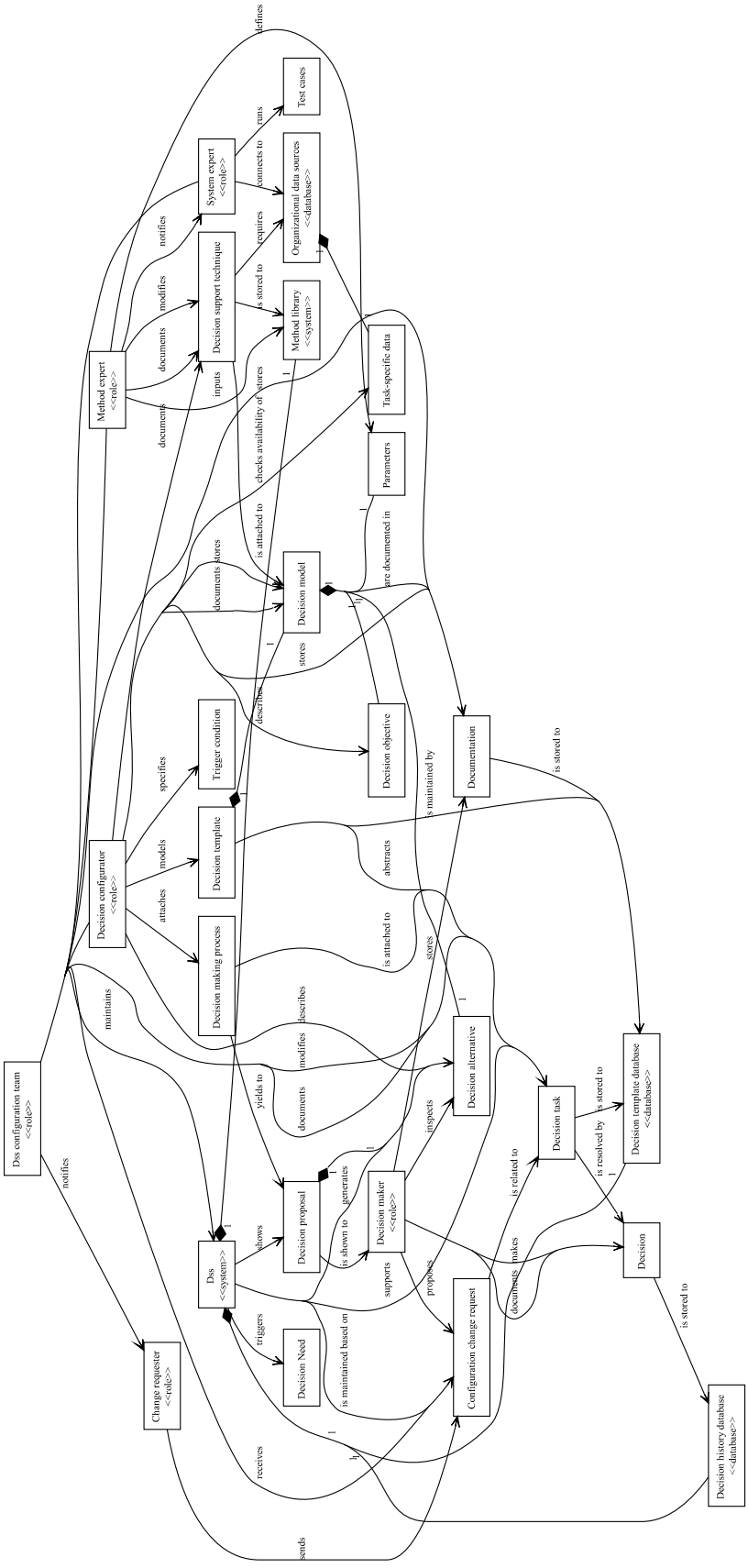

Figure 7.5. DSS entity model based on use cases – manual modifications applied.

7.4 Conclusion

Decision making processes are complex. There are a multitude of approaches and techniques to support decision making in organizations. We have tried to (re)cover all the relevant aspects of ODSS establishing linkage between themes described in the earlier chapters. This joins together the different roles and competences of the consortium project participants.

We have suggested both a new generic use case -based specification for operational decision support systems, as well as a way (stereotyped entity model) to establish a shared ontology between relevant stakeholders. Use cases were originally based on generic user requirements in Appendix A (Jokinen et al. 2008) and generalized in multiple iterations to accommodate different decision support techniques (e.g. statistical decision theory, data clustering). Use cases were expressed in ProcML format, allowing them to be published in a hyperlinked format and further processed by UCOT software. Although somewhat abstract in nature, the use cases clarify especially the organizational context (e.g. roles and information systems) needed to establish a decision support system. The semiautomatically generated entity model points out essential concepts from the problem domain and can be used as a base for more detailed specifications.

As usual on R&D&I, we have obtained results that point to further research. Although the use cases were based on generic user requirements, the explicit link between requirements and use case steps was not preserved. Even though ProcML supports linking requirements to use cases, as the meaning of particular steps were changed or as use cases were split or joined, tracing the original requirements to updated use cases was somewhat cumbersome without further software support. A more serious shortcoming is the lack of linkage between generated entity model and original use cases – the transformation is one-way and in most cases, manual corrections must be made to the entity model every time use cases are changed. In future development, the generated conceptual model should be synchronized with manually specified entities and stereotypes marked in use cases.

Combining use cases to semiautomatically generated, stereotyped entity model seems to be a promising approach for requirements elicitation and conceptual modeling regardless of the methodology (e.g. OOA/D, domain engineering, ontology engineering) used in later development phases. Stereotypes provide essential domain-specific metadata that can be used for code generation and simplify the original use case descriptions. Attaching a stereotype to each concept creates a classification of them, in this way supporting the critical transfer from domain analysis into system development. Some of the stereotypes (e.g. Role, System, Process, Document) are relatively domain-independent, but the exact method to derive different kinds of domain-specific stereotypes (e.g. DecisionModelElement) is yet to be explicated. Ultimately there could be transparent 2-way linking between requirements, use cases, and entities in a unified model residing in a knowledge base. Depending on the

modeling task, different views of the model could be exported to achieve significant productivity gains in systems development.

References

Abbott, R. J. (1983) Program design by informal English descriptions. Commun. ACM, Vol. 26, No. 11, pp. 882–894.

Arnott, D. (2006) Cognitive biases and decision support systems development: a design science approach. Information Systems Journal, Vol. 16, No. 1, pp. 55–78.

Bittner, K. & Spence, I. (2002) Use Case Modeling. Addison-Wesley.

Boyd, J. A. (2007) Discourse on Winning and Losing. http://www.d-n-i.net/dni/john-r-boyd/.

Cockburn, A. (1997) Structuring Use Cases with Goals. Journal of Object-Oriented Programming, Sept–Oct and Nov–Dec.

Cockburn, A. (2000) Writing Effective Use Cases. Addison-Wesley.

Davenport, T. H. (1993) Process Innovation – Reengineering Work through Information Technology. Harvard Business School Press.

Gorry, G. & Scott-Morton, M. (1971) A framework for management information systems. Sloan Management Review, Vol. 13, No. 1, pp. 55–71.

Jackson, M. (2001) Problem frames: analyzing and structuring software development problems. Addison-Wesley.

Jokinen, H., Grén, J., Hukki, K., Konkarikoski, K., Kärkkäinen, T., Pulkkinen, U., Ritala, R., Suominen, P. & Ylén, J. (2008) Generic User Requirements for Operational Decision Support System. *To be submitted*.

Kilpeläinen, T. (2007) Genre and ontology based business information architecture framework (GOBIAF). PhD thesis. Jyväskylä: University of Jyväskylä.

Kärkkäinen, T., Nurminen, M., Suominen, P., Pieniluoma, T. & Liukko, I. (2008) UCOT: Semiautomatic Generation of Conceptual Models from Use Case Descriptions. In: C. Pahl (Ed.) IASTED International Conference on Software Engineering (SE 2008). Calgary: ACTA Press. Pp. 171–177.

Lauesen, S. (2002) Software Requirements: Styles and Techniques. Addison-Wesley.

Mann, A. (2002) Risky Business. Columns (UGA Faculty Newsletter).

Nurminen, M., Honkaranta, A. & Kärkkäinen, T. (2005) ExtMiner: Combining Multiple Ranking and Clustering Algorithms for Structured Document Retrieval. In: Proceedings of International workshop on Integrating Data Mining, Databases and Information Retrieval (IDDI'05), 16th International Workshop on Database and Expert Systems Applications. IEEE. Pp. 1036–1040.

Nurminen, M., Honkaranta, A. & Kärkkäinen, T. (2007) ProcMiner: Advancing Process Analysis and Management. In: Proceedings of Workshop on Text Data Mining and Management (TDMM), IEEE 23rd Int. Conf. on Data Engineering. IEEE. Pp. 760–769.

Sharp, A. & McDermott, P. (2001) Workflow Modeling. Artech House.

Simon, H. A. (1965) The New Science of Management Decisions. In: Herbert A. Simon (Ed.) The Shape of Automation for Men and Management. New York: Harper and Row.

Simon, H. A. (1976) Administrative Behavior. 3rd edition. New York: The Free Press.

Soffer, P. & Hadar, I. (2003) Reusability of conceptual models: The problem of model variations. In: K. Siau, T. Halpin & J. Krogstie (Eds.) Proceedings of Eighth CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'03). Pp. 126–134.

Turban, E., Aronson, J. E. & Liang, T. (2004) Decision Support Systems and Intelligent Systems. 7th edition. Prentice-Hall, Inc.

Wirfs-Brock, R. (1993) Designing Scenarios: Making the Case for a Use Case Framework. The Smalltalk Report, Vol. 3, No. 3.

Zachman, J. A. (1987) A framework for information systems architecture. IBM Syst. J., Vol. 26, No. 3, pp. 276–292.

Appendix A: Generic user requirements for decision support systems

Generic user requirements for pre-structured decision tasks

GURs for generating and justifying decision proposals.

GUR label	GUR title	Description
GUR-1.1.1	Notify the user about a need to make a decision and act	Based on the measurements and models available, the DSS notices a situation that needs a decision to be made and brings this need to the user's attention.
GUR-1.1.2	Generate a proposal for a decision	Using data and models available, and by solving an optimization task a proposal is generated and presented to the user without any additional information.
GUR-1.1.3	Present the concep-tualization of system state, consequences and description of decision alternatives	Based on the measurements, system state descriptions and event history, the current system state is described with given concept system and all potential decision alternatives are presented in an understandable and acceptable form, and on request the consequences of user selected decision alternatives are presented.
GUR-1.1.4	Present measurement information relevant for decision to be made	The measurement data utilized in generating the decision proposal or elected by the user is presented; using available and suitable methods the uncertainty and reliability of the data is assessed and presented in an understandable and acceptable form.
GUR-1.1.5	Present the relevant state estimation and prediction models, their estimation and prediction results and uncertainties in them	Concerning the current decision proposal or user specified decision candidate, the relevant state estimation and prediction models are selected and visualized, and the produced estimation and prediction results with uncertainties are presented in an understandable and acceptable form.
GUR-1.1.6	Present the relevant objective(s) and the decision time horizon	The objectives used in the generation of decision proposal or user specified decision candidate, possibly by using a number of optimization time horizons, are presented to the user in an understandable and acceptable form.
GUR-1.1.7a	Present the degree of satisfaction of different objectives in multi-goal decision making	Given a decision proposal or user specified decision candidate the values of objectives and the level of satisfaction is presented to the user in an understandable and acceptable form. Also the measurement principle of the objective satisfaction is presented for the users. The motivation of the measurement principle is derived form the higher level objectives of the company.
GUR-1.1.7b	Present trade-off possibilities in multi-goal optimization	The trade-off ratios between the decision objectives are presented for the users in an understandable and acceptable form. One or several of the objectives may describe the attitude towards risk. The favored trade-off is motivated.
GUR-1.1.8	Show the robustness of proposed decision to user selected model *parameters*	Concerning the system state and prediction models used in generating the decision proposal, the sensitivity of this proposal is analyzed with respect to variations in parameters selected by the user. The sensitivity is visualized and presented in an understandable and acceptable form.
GUR-1.1.9	Analyze the robustness of proposed decision towards variations in user selected model *structures*	Concerning the system state and prediction models available in generating the decision proposal, the robustness of this proposal is analyzed towards variations in model structures selected by the user. The results of the robustness analysis are presented in an understandable and acceptable form.

GURs for modifying and developing a proposed decision.

GUR label	GUR title	Description
GUR-1.2.1	What-if analysis	The DSS allows the user to select alternative sets of measurement data and/or to change the parameters of state estimation models, prediction models and/or objectives to generate alternative decisions.
GUR-1.2.2	Manage the alternative decisions and their background material in a tree graph	The user is provided with an interface to manage alternative decisions and their background materials in a tree structure where a node is a fully structured decision task and a generated proposal and a link from one node to another specifies the change in decision task structure.
GUR-1.2.3	Facilitate group discussion about generating a consensus decision	The need and subjects for group discussion are noticed for users. The differences in initial structures are analyzed and a tree-like graph is generated from the initial structures. Through a process, a protocol and a template the group jointly modifies and develops the graph of decision structures further so that a consensus structures is specified. The graph documents the development of the consensus structure and the relationship between the initial structure and the consensus structure.
GUR-1.2.4	Facilitate group discussion about generating a decision by managing a hierarchy of alternative decisions and their background material, in particular conflicting objectives	The need and subjects for group discussion are noticed for users. The differences in initial structures are analyzed and a tree-like graph is generated from the initial structures. Through a process, a protocol and a template the group jointly modifies and develops the graph of decision structures without the objectives. Once a consensus structure has been achieved, a new decision task with the consensus structure complemented with all the initial objectives is formed. The decision proposal of this structure, the corresponding trade-off and level of satisfaction are analyzed in a joint session.

GURs for utilizing and storing experiences.

GUR label	GUR title	Description
GUR-1.3.1	Store a decision making session, link to future assessment	While the decision making session is carried out, the system stores all actions by the user so that the session can be rerun at any later time. As the user will make reference to process data in relative time, the session stores both the absolute time and relative time references to data so that the session can be rerun with original data or with the data of the rerun instant (see GUR-1.3.2). The user may specify future (over a user specified time interval) measurement data to be linked with the session. Such data would allow assessing the decisions that eventually were made and the system performance as a result of the decisions.
GUR-1.3.2	Retrieve similar decision making situations with link to follow-up (what really happened)	The DSS organizes the stored sessions for the end user in the order of similarity (measures of similarity: level 1 decision task; level 2 time since session; level 3 input data current vs. the one at the time of creation of session) and allows the user to rerun the session in one go or in steps with either the original data or present data.

Generic user requirements for structuring and analyzing decision tasks

GURs for structuring decision tasks.

GUR label	GUR title	Description
GUR-2.1.1	Specify condition for recognizing the need of making the decision	1) The user specifies a condition where some input from outside of the system is needed, i.e. decision making is needed. 2) An automated system can also be constructed to observe such situations when decision making is needed.
GUR-2.1.2a	Specification of system state space description	The user describes the specification of system state space with the possible aid from DSS. The specification concerns current decision task being structured.
GUR-2.1.2b	Specification of measurements available	The user lists the measurements available for the system and links the measurement names in data source to the names to be used in the DSS.
GUR-2.1.2c	Specification of information available	The user lists the information (*a priori*) available for the system.
GUR-2.1.3	Specification of decision consequence space description	The user describes the specification of decision consequence space with the possible aid from DSS. Specification includes the time horizon in dynamic optimization. The specification concerns current decision task being structured.
GUR-2.1.4	Specification of decision space	The user describes the specification of decision space with the possible aid from DSS. Specification includes the decision interval in dynamic optimization. The specification concerns current decision task being structured.
GUR-2.1.5	One-by-one specification of objectives as deterministic functions from consequence space to real numbers	The principle for defining measurement principle of the objective satisfaction for each objective is presented for the users (in a form of a template) The procedure for taking the higher level objectives of the company is described. The possibilities of different measuring principles are presented. The objective is specified as mappings from the consequence and decision space to real numbers
GUR-2.1.6	Specification of multiple and dependent objectives as deterministic functions from consequence space to real numbers	The principles for determining (additive) multi-objective value functions or multi-objective value models are presented. The objectives are specified as mappings from the consequence and decision space to real numbers.
GUR-2.1.7	Specification of attitude towards risk	The user specifies the attitude towards risk with assisted by the DSS. Descriptions may be based on utility, risk premium, or constraining the probabilities of unfavorable values of objectives.
GUR-2.1.8	Specification of inequality constraints in the decision space	The user describes the constraints in the decision space with assisted by DSS.
GUR-2.1.9	Specification of inequality constraints in consequence space	The user describes the constraints in the consequence space assisted by DSS. A constraint in consequence space can also be defined by constraining a specified objective.
GUR-2.1.10	Specification of other forms of constraints	Not all possible constraints are constraints entirely describable as those in decision space or consequence space. This requirement covers such additional constraints.

GUR-2.1.11	Specification of measurement information derivable from measurement data	The user specifies the forms of measurement information derivable from measurement data.
GUR-2.1.12a	Specification of state estimation model	The user has an access to a system database containing a set of state estimation models. The user selects a model and specifies the inputs to the model as measurement data or measurement information. The user may modify the structure of the model used.
GUR-2.1.12b	Specification of state recognition method	The user has an access to a system database containing a set of state recognition methods. The user selects a model and specifies the inputs to the model as measurement data or measurement information. The user may modify the structure of the model used.
GUR-2.1.13	Specification of consequence prediction model	The user has an access to a system database containing a set of consequence prediction models. The user selects a model and specifies the inputs as measurement data, measurement information or outputs of system state estimation model, and as decisions. The user may modify the structure of the model used.
GUR-2.1.14	Specification of optimization method to be used in the generation of decision proposal	The system has a library/database of optimization methods with documentation about in which case each of the methods is suitable, which are its parameters and instructions on how to choose them. The specification concerns current decision task being structured. This GUR requires that the user is educated concerning optimization methods.
GUR-2.1.15	A guided tour for structuring a decision task	The guided tour organizes the tasks corresponding to GUR-2.1.1-14 into a session that guarantees all the necessary definitions to be made for the decision task to be formally correctly structured. NOTE: as all well-structured decision support systems need not address all GUR-2.1.1-14 requirements, the guided tour has several exit points. The definition of well-structured problem is closely related to use cases supported.
GUR-2.1.16	Facilitate group work during the specification process	An approach for identifying the need of group work is provided. Support for identifying sufficient set of participants, and their roles for the group work is given. The system supports the specification tasks of GUR2.1.1-14 or the guided tour of GUR2.1.15 to be carried out in a joint discussion sharing the tool and the structures over the network.
GUR-2.1.17		
GUR-2.1.18	Maintain long term history of decision support structures	Version management of decision structures with documentation.
GUR-2.1.19	Allow to document the choice of structures	Generates a structured document documenting the decision support structure. Can be filled during the specification process or once the structure is fully defined. Mainly for documenting systems set up for permanent use, but can be used also in documenting ad hoc decision making.
GUR-2.1.20	Manage the portfolio of structured decision tasks	A user interface for the database of supported structured decision tasks. At first level lists the supported tasks. At second levels describes the task structures. At third level allows access to all recorded decision sessions. The user may select existing structured decision tasks as a basis for generating a new structured decision task. The user interface supports virtual group work on the tasks.

GURs for tuning parameters for decision tasks.

GUR label	GUR title	Description
GUR-2.2.1	Present all the decision support parameters to be set	The user is provided with a list of all relevant parameters needed to be set in tuning the support for a decision making task.
GUR-2.2.2	Provide an interface to set all the decision support parameters	The user is given an access to modify the decision support parameters.
GUR-2.2.3		
GUR-2.2.4	Support identifying parameters in state estimation, consequence prediction and objective functions with history data	In order to find a set of parameters for the decision task, the user may use history data to identify the state estimation and prediction model parameters.
GUR-2.2.5	Support testing the DSS with user specified data	The user may shift the present time to some earlier instant about which history data is available. For data not available through history data base, the user may specify fictitious data, e.g. through excel sheets or as program expressions.
GUR-2.2.6	Manage the alternative sets of decision support parameters and their test results	The user is provided with a user interface for the database to store and retrieve any alternative sets of decision support parameters and their test results.
GUR-2.2.7	Support tuning and testing of parameters as group work	Provides a format for making expert judgments for the values of parameters. Judgments can be combined and uncertainties assessed. The system supports the tuning tasks of GUR-2.2.1-6 to be carried out in a joint discussion sharing the tool and the structures over the network. Each group member may work individually to identify a subset of parameters. DSS will combine such individual pieces of work to a single set of parameter values or their alternative values to be further discussed within the group.
GUR-2.2.8	Maintain long term history of tunings and their tests	Shows the history of structure describing the decision task. Allows reverting to some any earlier version.
GUR-2.2.9	Allow to document the choice of parameters	Allows inputting text to explain why a particular value of parameter has been chosen.