A Task-oriented Requirements Engineering Method for Personal Decision Support Systems
A Case Study

Christian Kücherer[1] and Barbara Paech[1]
[1]*Institute for Computer Science, Heidelberg University, Im Neuenheimer Feld 205, 69120 Heidelberg, Germany*
{kuecherer|paech}@informatik.uni-heidelberg.de

Keywords: Decision Support System, DSS, Personal DSS, Requirements Specification, Task-oriented Requirements Engineering, TORE

Abstract: **[Context and motivation]** Personal Decision Support Systems (PDSSs) are information systems which support executives in decision-making by a decision- and user-specific data presentation. PDSSs operate on current data with predefined queries and provide a rich user interface (UI). Thus, a Requirements Engineering (RE) method for PDSSs should support the elicitation and specification of detailed requirements for specific decisions. However, existing RE approaches for decision support systems typically focus on ad-hoc decisions in the area of data warehouses. **[Question/problem]** Task-oriented RE (TORE) emphasizes a comprehensive RE specification which covers stakeholders' tasks, data, system functions, interactions, and UI. TORE allows an early UI prototyping which is crucial for PDSS. Therefore, we want to explore TORE's suitability for PDSSs. **[Principal ideas/results]** According to the Design Science methodology, we assess TORE for its suitability of PDSS specification in a problem investigation. We propose decision-specific adjustments of TORE (DsTORE), which we evaluate in a case study. **[Contribution]** The contribution of this paper is threefold. First, the suitability of the task-oriented RE method TORE for the specification of a PDSS is investigated as problem investigation. Second, a decision-specific extension of TORE is proposed as the DsTORE-method in order to identify and specify details of decisions to be supported by a PDSS. DsTORE is evaluated in a case study. Third, experiences from the study and method design are presented.

1 INTRODUCTION

Today, data warehouses (DW) are standard technology for decision support in companies. They are a specific form of Decision Support Systems (DSSs) which represents a subgroup of information systems. As discussed in a recent overview by Hosack et al. (Hosack et al., 2012), there is a large variety of DSSs and an extensive history of research on DSSs. At the same time, this research is ongoing and incorporates new trends such as social media or mobile computing (Gao, 2013). Similarly, there is a growing field of research focusing on the development methodologies for DSS. Saxena (Saxena, 1991) is a very early development methodology which emphasizes the understanding of decision tasks and prototyping. In a more recent review Gachet and Haettenschwiler investigate early and widely acknowledged development processes of DSSs (Gachet and Haettenschwiler, 2006), focusing on either decisions or system engineering or both. They also emphasize the importance of an evolutionary approach which pays attention to both, organizational and technical issues. In the same vein Arnott (Arnott, 2010) develops a business intelligence development approach starting from a fundamental understanding of senior executives' behavior. We are interested in Personal DSSs (PDSSs, introduced in detail in Section 2.2) which are small-scale systems that support decision-making of one or a small group of executives with an exactly defined set of supported decisions (Arnott, 2008). While the early approaches to DSS provide a general framework, they do not describe a detailed requirements engineering method. Whilst retaining the emphasis of these methodologies, we take the task-oriented RE approach TORE (Paech and Kohler, 2004; Adam et al., 2009) as our basis to develop a detailed requirements engineering method for PDSSs. TORE has proven useful for information systems and it supports a refinement of requirements concerning all system details up to the UI, and thus to early prototyping. However, it has not yet been applied to PDSSs.

Table 1: Operational and decisional systems. (Left three columns acc. (García et al., 2016) and (Salinesi and Gam, 2009), except rows *structure of data* and *predefined decisions*)

	Operational	**Decisional**	**PDSS**
Objective	busin. operation	busin. analysis	specific busin. decisions
Main functions	daily operations (OLTP)	DSS (OLAP)	decision-specific UI data presentation, reporting
Decisions	n/a	un-,semi-,structured	semi-, structured
Usage	repetitive, predef.	innovative, unexp.	repetitive, predef.
Design orientation	functionality	subject	subject
Kind of users	clerk	executives	executives
Number of users	thousands	hundreds	one to ten
Accessed tuples	hundreds	thousands	hundreds
Data sources (dsrc)	isolated	integrated	heterogeneous, isolated
Structure of dsrc	structured	structured	un- and structured
Granularity	atomic	summarized	atomic and summarized
Time coverage	current	historical	current and historical
Access(work units)	simple transact.	complex queries	simple queries
Size	MB/GB	GB/TB/Petabytes	MB

With this recent scholarly background in mind, the overall Research Question (RQ) of this paper is: *How can TORE be extended to support the RE of a PDSS?*. We follow the design science cycle described by Wieringa (Wieringa, 2014) to answer this RQ. In the *problem investigation*, we evaluate TORE's suitability to support the RE for PDSSs. In the *treatment design*, we propose a decision-specific adaptation of TORE (DsTORE). Finally, in the *treatment validation* we evaluate DsTORE.

This paper is structured as follows. Section 2 introduces PDSSs, presents the case for this study, and offers a brief overview of TORE. Section 3 discusses related work on RE for DSSs. The investigation of TORE's support for the RE of PDSSs is presented as problem investigation in Section 4. The treatment design DsTORE is presented in Section 5. The treatment validation is described in the form of a case study in Section 6. The results of the problem investigation and case study are discussed in Section 7, in which we also present the lessons that we have learned. The threats to validity are discussed in Section 8. Finally, the paper is concluded in Section 9 by sketching some ideas of future work.

2 BACKGROUND

This section presents a brief introduction of PDSSs and task-oriented RE as well as the case for our case study.

2.1 Decisions and Decision Knowledge

A decision can be defined as a choice that is made between multiple alternative courses of actions, and that in turn leads to a desired objective (Holsapple, 2008a). Alternatives have implications and decision-making requires knowledge. For example, the prioritization of projects (choice) results in a precedence of one project over others (alternatives) based on a project's importance, benefit for operations, or the estimated budget (knowledge). *Structured decision problems* have defined criteria in order to make the decision, known alternatives and implications, and require specific types of knowledge. By contrast, *unstructured decision problems* are by their nature unexpected, have unstable context and unknown alternatives, possess unknown criteria, and rely on situations in which none or an incomplete level of knowledge is available (Holsapple, 2008a). *Semi-structured decision problems* are in between. Decision knowledge is taken from a *data source*, persisting data for future access via interfaces or file access. Examples of data sources are databases or documents such as spreadsheets and text documents. A data source is described through data source format, location, and content. For example a project list used by an executive can be described using a spreadsheet (format), a URL (location), and a list of projects (content). This is in contrast with unstructured data, in which structured data follows a formal definition of data types. Thus, data sources of structured data are typically databases, while semi-structured data is held in spreadsheets, and unstructured data in text-documents.

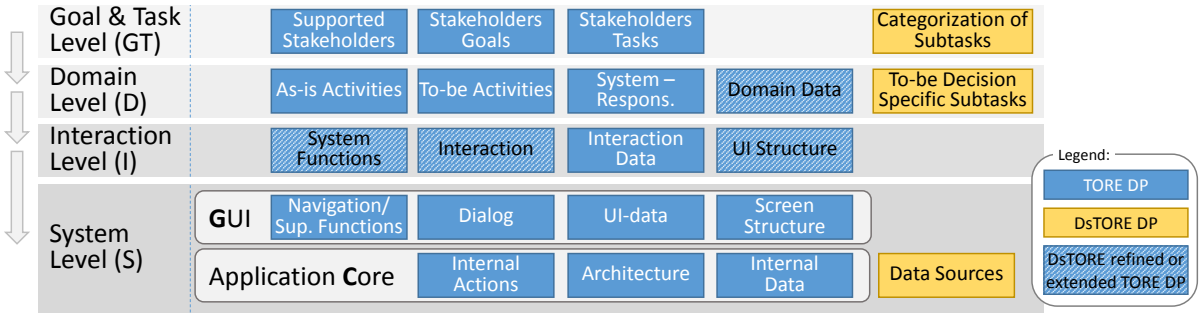

Figure 1: Decision Points of the TORE Framework (Paech and Kohler, 2004) and PDSS-specific extensions

2.2 Personal Decision Support Systems

In general DSSs support all kinds of decisions. Salinesi et al. (Salinesi and Gam, 2009) provide a comparison of operational information systems (IS) and DW (called decisional IS) which is adapted in (García et al., 2016). DSSs in general cover a broad range of different system types and incorporate a variety of different techniques such as decision models or artificial intelligence (Holsapple, 2008b; Arnott and Pervan, 2005). The specifics of Personal DSSs have been described by Arnott (Arnott, 2008). He summarizes that PDSSs are developed for one or a small group of managers with the goal to improve the process and outcome of decision-making. The scope of a PDSS is the support of one or a small number of decision tasks, where often semi-structured data based on spreadsheets are used. The fact that PDSSs are small-scale systems refers to the supported small number of users and features and requirements. To delineate PDSSs from DSSs in general, we add a third column to Table 1. The *objective* of PDSSs is to support decision-making within a defined context. Its *main function* is to provide specific and predefined information for structured and semi-structured decisions in an overview UI or as a report to executives. In a similar way to DW, the primary *users* are executives, who access heterogeneous and isolated *data sources* at runtime. In contrast to DW, usage is repetitive and predefined and the amount of accessed data is not extensive. Altogether, the data and data sources of PDSSs are quite complex, while their usage is rather simple.

2.3 Task-oriented Requirements Engineering

Task-orientation focuses on the RE process and helps to deliver software that satisfies user needs (Paech and Kohler, 2004). As a conceptual framework, TORE provides a conceptual model for RE and has been successfully applied in a number of IS development projects of different problem domains (Adam et al., 2009). Important aspects of the requirements specification are determined in each of TORE's 18 Decision Points (DPs), which are grouped into four levels of abstraction, as shown on the left part of Figure 1. Note that TORE's DPs are distinct from *decisions* in the context of DSSs. The goal & task level focuses on stakeholders, goals, and tasks. Tasks are often activities within larger business- or management-processes, but the latter of these is outside of the focus of TORE. The domain level accommodates as-is and to-be activities which refine the tasks by subtasks that contribute to the completion of a stakeholder's task. Further, this level contains system responsibilities (often called system features), and domain data. The interaction level determines how stakeholders will be supported in their to-be-activities by the system. Finally, the system level determines UI details and infrastructure including the system architecture. TORE does not fix or prescribe artifact types for documentation of the DPs. In particular, it is not necessary to have a separate artifact for every DP, as lower-level DPs contain decisions for corresponding higher-level DPs. Several artifact types have often been used, as is shown in Table 2 by their relation to the DP. The artifact type **task description** according to Lauesen's Task&Support (Lauesen, 2005) describes stakeholders' tasks. **Subtask templates** are used to describe the DPs As-is and To-be-activities. The DP **Domain Data** is described with an UML class diagram or entity relationship diagram, containing domain entities and their relationships. The **UI Structure Diagram** shows the navigation between workspaces for DP UI Structure, where **workspaces** group system functions. **Virtual windows** (Lauesen, 2005) describe the DP Screen Structure by an high-level grouping and structuring of the UI data in order to illustrate the information needs for tasks early on in the RE process. The **UI Prototype** refines the virtual window with the navigation functions and dialogue and documents the DP UI Data, Dialog and Navigation Functions.

Table 2: TORE Decision Point and supported artifact types relevant for this study (Paech and Kohler, 2004). Rows marked in grey are introduced in Section 5.

Level	Decision Point	Artifact
GT	Stakeholders Tasks	Task description
D	As-is activities, To-be activities	Subtask template
D	Domain Data	UML class diagram, ER diagram
D	Categorization of ST	Task description
D	To-be Decision specific ST	Decision specific subtask template
I	UI Structure	UI structure diagram
I	System functions	Function description
I	Interaction Data	UML class diagram
I	Interaction	use case text
G	UI Data	UI Prototype, Virtual window
G	Screen Structure	UI Prototype, Virtual window

2.4 The Case and Its Context

The case was selected on the basis of availability as part of the research project Semantic Network of Information Management in Hospitals (SNIK) (Jahn et al., 2014). The project team develops, among other things (Schaaf et al., 2015), a dashboard for the CIO of a hospital called CIO-Navigator (CION). The CIO is the head of the Information Management (IM) Department and is responsible for decision-making in strategic, tactical, and operational IM. Thus, the CIO role is located at management or executive level. The CIO is currently faced with the problem that he struggles with distributed and scattered information in several systems and documents. He requires this information to make decisions related to the IM and for each decision the CIO has to collect the necessary data. By the use of CION, we enable the CIO to navigate current data of the IM department and provide him with the relevant set of information required for decision-making of a predefined set of recurring decisions. According to the characteristics presented in Table 1, CION is a PDSS.

3 RELATED WORK

For related work, we first discuss two papers which provide an overview of RE for DSSs and then we discuss goal-oriented RE approaches for DSSs.

Garcia et al. (García et al., 2016) present a comprehensive mapping study of 27 articles related to RE for DSS. Although the title suggests DSSs, the search terms and the identified literature focus on DW. The authors conclude that there is a gap in the process of creating the design of a DSS in a methodical manner. In particular the business needs gain too little attention during the RE process. They propose to elicit business needs from stakeholders by asking "what do you do and why?". Another overview by Salinesi and Gam (Salinesi and Gam, 2009) (not covered by Garcia et al.) provides a comprehensive discussion of 15 requirements engineering approaches for DSSs, all of which are DW-specific. They found three families of methodological processes. (1) Data-driven approaches which are focused on the structure of data sources to design multi-dimensional schemas. However, these approaches do not cover decision specifics. (2) Requirement-driven approaches which focus on decision makers' requirements. The weakness of these approaches is that the availability of data sources is not treated sufficiently so that user requirements may not be realizable. (3) Mixed-driven approaches are a combination of (1) and (2). Both overviews again confirm the need to understand the business and stakeholder needs.

As the approaches of both overview papers focus on DW, they support in particular the specification of DW data schema definitions, data marts for DWs, and DW-specific ETL (extract, transform, load) processes, which are not important for PDSSs. Thus, these new RE approaches also do not provide a detailed task-oriented guideline for PDSS development.

There are several goal-oriented RE approaches for DSS. Again they focus on DW. They share the importance of the business and decision focus, but focus on decision specifics for DW such as key performance indicators (Pourshahid et al., 2014) or business strategy and indicators (Barone et al., 2012; Topaloglou and Barone, 2015). Giorgini, Rizzi and Garzetti propose with GRAnD (Giorgini et al., 2008) a goal-oriented method which connects a decisional model with a source schema. While this method provides a detailed analysis of the goals and decisions, it does not provide guidance for the UI design and early prototyping with the user.

Altogether, we did not find a requirements engineering method for DSS in general and PDSS in particular which provides guidance for a seamless transition from the business level to UI details.

4 PROBLEM INVESTIGATION

This section describes the problem investigation, in which we wanted to understand what decision specific information is missing in TORE to support the

specification of a PDSS. In particular, we were interested in those decision details that are necessary for PDSS requirement specification yet inadequately represented in TORE's DPs.

4.1 Design and Data Collection

We investigated the RE process for the specification of CION (cf. Section 2.4) based on the method TORE. The supported stakeholder (TORE DP Supported Stakeholder) is the CIO of a particular hospital. Firstly, we performed a pre-assessment of the tasks and data of the IM department by a combined task- and system analysis (Ammenwerth et al., 2015) to gain an overview of the tasks and responsibilities of the CIO and IM department (Kücherer et al., 2015). The goal of the CIO is to provide a transparent view of the operative and tactical data of the IM department for IT-staff and visitors (documented in the TORE DP Stakeholders Goals). The resulting artifact was a task description according to Lauesen's Task & Support containing several CIO tasks. In consultation with the CIO, we prioritized the tasks depending on their importance and selected the first two tasks *project management*, and *change management* for further investigation. These two tasks were further detailed by a semi-structured interview (duration 2:15) with the CIO which helped us to deepen and refine our understanding of TORE's DP Stakeholders Tasks and Domain Data wrt decisions. The refined task description and the meeting minutes were reviewed by the CIO.

During a document analysis based on screenshots, we analyzed two spreadsheets provided by the CIO in order to understand their decision specifics. These screen shots contained the majority of data necessary for the investigated task project management and change management. In an additional document analysis based on operative files, we analyzed further decision-specific documents in the form of spreadsheets (cf. Figure 3) and text-documents. The document analyses helped us to understand particularly the decision-specific data. We did not visit the other DPs, as the insights from these activities already provided a good picture of the decision-specific requirements.

4.2 Results

The problem investigation shows that TORE's DP can capture a large amount of relevant requirements, which have been gained during the interview. Some details of important information about the CIO's decisions cannot be captured explicitly with the existing artifact types and DPs in TORE, explained in the following. We identified five decision-specific pieces

ID: Subtask	PM 3a: **P**rioritize projects of `project list`
Actor	CIO
Contribution	Consider important projects in budget planning.
Cause	Next fiscal year starts in two months
Description	Decision about order of new projects for budget planning. Necessary step before a project can start. The assigned priority depends on the project contribution...
Pre condition	all project proposals for next fiscal year are available. CIO knows goals and necessity of projects
Info-In	urgency, proposed priority of proposer (project proposal), project list, controlling-list, remaining budget (difference calculation).
Post condition	Changed priorities in project list. Budget is transferred to new project.
Info-Out	Priority of new project. Reprioritization of existing projects in project list.

Figure 2: Subtask `Project Prioritization` in standard template

of information which we postulate as criteria c_1-c_5. These criteria will shape the treatment design in the next section.

c_1: *Distinction of decisions from conventional subtasks*. We found subtasks with and without decision character. Some subtasks contain a decision problem as a recurrent choice between several alternatives, each of which entails different implications (cf. example in Sec. 2.2). Such subtasks are in particular relevant for a PDSS requirements specification.

c_2: *Detailed description of the data necessary to make the decision*. Based on the information from the pre-assessment and interview, we created the subtask description `prioritization of projects` as a subtask of the task `project management`, as is shown in Figure 2. The documentation with a standard subtask template does not show explicitly the data required by a decision. The domain data model shows the entities, but not the relation between entities and decisions. By way of an example, Figure 2 shows two deficits: It cannot be explicitly documented, (a) how the remaining budget is calculated, i.e. which entity-attribute pairs are used for the calculation, and (b) which information exactly is necessary from the `project list` and the `controlling list`.

c_3: *Decision-specific rules to choose from alternatives*. Decisions always contain alternatives (cf. Section 2.2). In some cases, the alternatives can be chosen based on predefined rules. To understand decisions and their alternatives, it is essential to emphasize rules, if there are any. For example, the CIO explained the rule to assign a priority to new projects.

Running Projects total: 2		Year 2016			95.000 €	50.000 €	90
Prio	**Project ID**	**Title**	**Description**	**planned start**	**planned budget**	**assigned budget**	**HR**
1	15-1	Server Virt.	Migration ...	Q3/2015	120.000 €	90.000 €	260
	16-17	Letter softw.	Update phy...	Q4/2016	15.000 €	0 €	25
2	16-3	Service Mngr	Add KPI ...	Q2/2016	80.000 €	50.000 €	65

(a) Part of `Project List`

Account ID <ID>		Planned Service Total	120.000 €	7.544 €
Project lead	**Project ID**	**Title**	**Assigned Budget 2016**	**Remaining budget**
J.Doe	15-1	Server Virtualization	40.000 €	21.000 €
A.Smith	16-17	Letter software		0 €
H.Simpson	16-3	Service Manager	80.000 €	-13.456 €

(b) Part of `Controlling List`

Figure 3: Decision-specific Spreadsheets

Projects contributing to service protection receive priority one, contributions to service operation receive priority two, and all other projects receive priority three. This rule is not made explicit in Figure 2.

c_4: *Support of decision-specific patterns in data for tables and summary fields*. The document analysis has shown two reoccurring patterns in the data that is required by the CIO to make a decision. First, combinations of entity/attribute pairs from more than one single entity are evident. Second, computed data such as sums or any other kind of condensed data is evident. For instance, the CIO needs to prioritize new projects, which might change the priority of existing projects. Currently, for the decision regarding the priority of new projects the CIO requires two documents which he presented to us after the interview: (1) The spreadsheet `project list`, illustrated in Figure 3(a) contains all running, finished, and planned projects. Each project is specified with a title and priority, a planned start date, the estimated budget and Human Resources (HR) effort, and an approved budget. In the top row, there is the number of projects and the sum of planned and assigned budget for projects running in year 2016. (2) The `controlling list` which contains an overview of all projects' budget status as shown in Figure 3(b), based on each project's expenses (not shown here). The list contains the project's leader, ID, and title, as well as the assigned budget in 2016 and the remaining budget. Again, there are single rows and summary fields in the header, such as the total assigned budget in 2016 and the total remaining budget (`unused_budget`) of all projects. It is important to describe explicitly and model these decision-specific data patterns (besides in UI prototypes), in order to be able to aggregate all data in one model and check for dependencies and consistencies. The domain data model does not support the modeling of data combinations from different entities or aggregations.

c_5: *Relation between content, format, and URL for data sources*. The problem investigation has shown that executives work with spreadsheets containing aggregated data (e.g. Figure 3(a), 3(b)) and some text-documents. Their content and format is known to frequent users. Spreadsheets and text-documents represent heterogeneous decision-specific data sources. In order to relate the patterns in data (c_4) with their origin, we need to document and understand these data sources. As explained in Section 2.2 this heterogeneity is specific to PDSSs. Thus, a detailed description is an important input for system design decisions. Such a description should contain the content in terms of entities, the format and the URL.

5 DSTORE - TREATMENT DESIGN

The design of the DsTORE method addresses the design problem: *Improve the TORE method by decision-specifics c_1-c_5 in order to gain a comprehensive and structured specification for PDSSs.* Thus, the emphasis is to provide a more precise RE. This section presents a short introduction of DsTORE as TORE enhancement. In this stage, we first introduce new decision-specific DPs with related artifact types (cf. the right side of Figure 1). In some cases, we adapt DPs, indicated as dashed DPs in Figure 1. DsTORE uses the same artifact types as TORE. There are two extended artifact types, firstly the subtask description and domain data model, and secondly one new simple data source model.

5.1 Goal & Task Level

The new decision-specific DP **Categorization of subtasks** that fulfills criterion c_1 is required to analyze subtasks with the help of a categorization scheme in order to find subtasks which *are* decisions. Two properties are used for categorization: the presence of alternatives to choose from and the verb or substantive of the subtask, indicating a decision. Subtasks can be a *decision* or a *conventional subtask*. For the IM domain, we use a non-exhaustive model encompassing 7 categories and distinguish objects or situations (x) and subjects(y). In other domains, the categorization scheme needs to be populated differently. Decisions are **A**pproval of x, **E**valuation of x or y, and **P**rioritization of x. Conventional

ID: Subtask	PM 3a: **P**rioritize projects of *project list*		
Actor	CIO | **Supporting Actors:** project leader		
Contribution	Consider important projects in budget planning.		
Cause	Next fiscal year starts in two months		
Description	Decision about order of important projects...		
Pre condition	all project proposals for next fiscal year are available. CIO knows goals and necessity of projects		
Combined Data	**Attribute**	**Entity**	
	project.priority	Project List Table Entry	
	project.start	Project List Table Entry	
	...	...	
	budget.planned	Project List Table Entry	
Computed Data	**Attribute**	**Entity**	**Computation**
	budget.remaining	Controlling List Table Entry	assigned budget - sum(project expenses)
	<derived attribute> unused_budget	Controlling List	sum(budget.remaining, all projects))
Post condition	Defined order, assigned priority of new projects. Budget is transferred to new project.		
Info out	**Attribute**	**Entity**	
	project.priority	Project List Table Entry	
Rules	Priority definition by project contribution: (high) service protection; (medium) service operation; (low) nice to have.		

Figure 4: Decision `Project Prioritization`

subtasks are **M**onitoring of *x*, **D**ocumentation of *x*, **C**ommunication of *x* to *y*, and **S**upport of *x* or *y*. Examples are the conventional subtask 'monitoring of project progress' classified as M and the decision 'approval of budget' with the alternatives rejection or approval classified as A.

5.2 Domain Level

Decision-related subtasks are specified in the DP **To-be decision-specific subtasks** and are documented with the decision-specific subtask template. The *decision-specific subtask template* extends TORE's subtask template to define the semantics for decision artifacts and fulfills c_2 and c_3. It covers details of a decision in four structured fields, which are marked yellow in Figure 4. *Combined data* express necessary decision- and/or stakeholder-specific data with entity/attribute pairs. Data aggregations, e.g. of time or spatial data often used for decision-making, are documented by *computed data* and indicated by '/' for derived attributes in the Domain Data Model (cf. Figure 5). Details of the computations, such as add, subtract, etc., are documented in the column `computation`. The result of the decision is documented in the field *Info-out*, again with entity/attribute pairs. *Rules* describe which alternative will be chosen by the decision maker and under which circumstances. The notation can be textual or formal, as is appropriate. Decisions require specific domain data that is decided in the extended DP **Domain Data** which uses additional stereotypes.

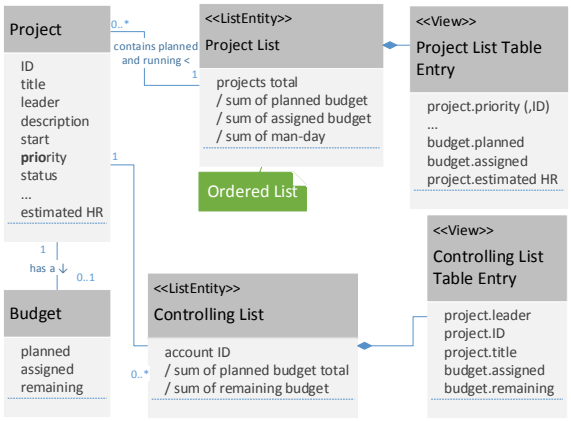

Figure 5: Part of Domain Data Model for `Project Management`

The attributes and entities need to be consistent with those of the decision-specific subtask. The decision-specific spreadsheets, as presented in the problem investigation (cf. Section 4), are modeled in the domain data with UML stereotypes. Lists, such as the project list (cf. Figure 3(a)), are modeled by stereotype «`ListEntity`». The «`ListEntity`» is related to the listed entities and to a «`View`» entity which captures combined attributes. As an example, both, the `controlling list` (cf. Figure 3(b)) and the `project list` (cf. Figure 3(a)) are modeled in the domain data model in Figure 5. The class «`ListEntity`» `Controlling List` holds computed and non-repetitive attributes such as the `sum of planned budget` and the `sum of remaining budget` (i.e. the `unused_budget` of other projects) which are located at the top of the list. The table rows are modelled by the class «`View`» `Controlling List Table Entry`. Classes of «`View`» might contain attributes of several domain entities, e.g. the ID of a project as attribute `project.ID` and the remaining budget of the same project as attribute `budget.remaining` for `Controlling List Table Entry`.

5.3 Interaction Level

In the DP *system functions*, TORE focuses on the to-be system functions. DSSs have a limited range of typical system functions, cf. Holsapple (Holsapple, 2008b), and Power (Power, 2011). Therefore, we add criterion c_6: the RE specification should consider typical PDSS-specific system functions explicitly. Examples of PDSSs system functions include creation of reports, navigation to data sources, and document export. We adapt the DP to use a decision-specific predefined set of system functions as a guidance to

Table 3: Overview of the Case Study's activities, duration, and artifact creation effort. (T=Telephone), Duration in [h:mm]

ID	Dur.	Activity	**Decision Point** / Description
Tasks of Treatment Validation Phase 1: project management, change management			
1	1:20	Interview	Categorization of Subtasks, To-be Decision Specific Subtask, UI Structure
2	1:20	Interview	System Functions, Interaction Data
3	T1:30	Interview	System Functions, Interaction Data, UI Data
4	1:45	Interview	To-be activities, Domain data, Screen Structure
5	-	Development	Development of system prototype
6	1:00	Presentation	System prototype in software
Tasks of Treatment Validation Phase 2. Task: IT Strategy			
7	1:30	Interview	Stakeholders Tasks, Categorization of Subtasks, To-be Decision Specific Subtasks, Domain Data, Interaction Data
8	1:30	Interview	To-be Decision Specific Subtasks, Interaction Data
9	T0:45	Evaluation	Eval.: interviews, artifacts, system prototype

support completeness and to avoid gold plating. This set can be adapted according to the different types of DSS. We also adapt the DP Interaction which is typically described by use cases. Since PDSSs provide a dashboard-like UI with one screen for each decision, a detailed description of interaction on this level is not necessary. The UI-structure diagram suffices to capture the navigation between the different workspaces.

5.4 System Level

The DP UI-Data and Screen Structure is documented with a Virtual Window. Figure 6 shows the Virtual Window of the subtask *prioritize projects of project list* (as introduced in Figure 4). The Virtual Window shows the arrangement of the decision-specific subtask's data. On the system level, there is a new DP **Data sources**. In TORE, data sources are left to software design. As argued for c_5 the consideration of data sources in DsTORE is important. Data source description can be captured in a simple table with the rows *entities*, *format*, and *location*, e.g. `project`, `excel/csv`, `http://filesrv1/2016/project-list.xlsx`.

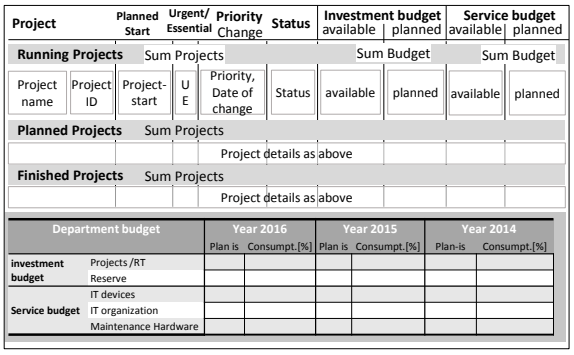

Figure 6: The virtual window for `Project Management`

6 CASE STUDY: TREATMENT VALIDATION

This section describes the design and results of the two-phase case study to evaluate the treatment design DsTORE. The case and it's context is described in Section 2.4, since it was relevant for the problem investigation.

6.1 Design

The research objective is to identify the extent to which DsTORE supports the RE of PDSSs. The *object of study* is the application of the DsTORE method. Therefore, we raise *RQ.1: How well does DsTORE support the specification of requirements for a PDSS*? In **phase one** we re-visted the tasks project management and change management (already investigated in the pre-assessment), but now using DsTORE artifact types for specification. Missing information was additionally elicited. We used four semi-structured interviews (activity 1-4) to elicit the requirements with DsTORE, as shown in Table 3. DPs visited are: Categorization of subtasks, As-is and to-be, To-be decision-specific subtasks, UI Structure, System functions, Interaction data, and Screen Structure. Table 5 shows which artifacts have been created after the elicitation during the activities 1-4 and 7-8. For each artifact the time needed for its creation is given, including an internal review and a review with the stakeholder. The task description of project management and change management were used from the problem investigation. All other artifacts were newly created.

The DP Navigation/Supporting Functions was not relevant, since no supporting functions were necessary. The DP Dialog was not relevant, since the dashboard does not support complex dialogues. All sys-

tem DPs were refined during the implementation of the system prototype in activity 5, where two students implemented a system prototype in .NET which integrates into SharePoint. We presented this system prototype to the CIO in activity 6. In **phase two** we investigated the CIO task IT-Strategy with *activity 7-8*. The DPs Stakeholders Task, Categorization of subtasks, To-be decision-specific subtasks, Decision-specific domain data, and Interaction data have been visited in activity 7. In activity 8 we refined the requirements of DPs To-be decision-specific subtasks, and Interaction data.

Table 4: Metrics of data collection for the case study - treatment validation

ID	Description of metrics
m1	Time taken for the creation of artifacts
m2	Number & duration of interview sessions in total
m3	Stakeholder's feedback to the interviews
m4	Stakeholder's feedback to the artifacts
m5	Stakeholder's acceptance of the system prototype
m6	Requirements engineer's feedback

We performed the data analysis for both phases after all interviews. To answer RQ.1, we used a Goal Question Metric approach (GQM) (Van Solingen et al., 2002). We study the following effects of the application of DsTORE (see (Wieringa, 2014) for similar distinctions): the **efficiency** in terms of the time taken for artifact creation and the interviews (m1 and m2), the **usability** in terms of the ease of use of the artifacts and the process for the stakeholder (m3 and m4) and the requirements engineer (m6), and the **utility** in terms of the value of the outcome for the stakeholder (m5). The data for these metrics was collected in a semi-structured interview (activity 9). Table 4 summarizes the metrics used to answer RQ.1.

6.2 Results

The creation of artifacts (m1) took 106 hrs in phase one and 7 hrs in phase two, cf. Table 3. Four interviews with a duration of approx. 7 hrs. were conducted (m2) in phase one, and two interviews with a duration of 3 hrs. in phase two. The CIO liked the interviews of both phases and rated the semi-structured execution of interviews as good (m3), since it gave him room to discuss ideas. He suggested two improvements: the provisioning of a RE-process description with a graphical sequence upfront, and the consultation of other departments' experts in some situations. The CIO found it sometimes difficult to provide all relevant decision information. It would have helped him, if he had a prepared description of decisions before the RE phase. The meeting minutes

Table 5: Overview of artifacts created (ID = Activity ID referring to Table 3, ST=Subtask)

ID	Decision Point	Artifact	Dur.
1	Categorization of ST Decision specific ST UI Structure Screen Structure	Task description decision-specific ST UI structure diagram Virtual Window	56 h
2	System functions Interaction Data	function description Domain Data Model	15 h
3	System functions Interaction Data UI Data	UI structure diagram Domain Data Model Virtual Window	23 h
4	Interaction UI Structure Screen Structure	UI Prototype UI structure diagram Virtual Window	12 h
7	Stakeholders Tasks Categorization of ST Decision specific ST Domain Data Interaction Data	Task description Task description decision-specific ST Domain Data Model Domain Data Model	3 h
8	Decision specific ST Interaction Data	decision-specific ST Domain Data Model	4 h

Table 6: Metrics and CIO's artifact rating
(• difficult, ○ easy, ○○ very easy, ++ very important, + important, - less important)

Artifact type	simplicity		import.
	understs.	comment	understs.
Task description	○	○	+
Categorization of subtasks	•	•	+
Decision Specific Task Description	•	○○	+
Data sources	○○	•	++
Workspaces	○	•	-
Virtual windows	○○	○	++
UI Structure Diagram	•	•	+
UI Prototype	○○	○○	++

and the correction iteration was good and helpful for the CIO. The interview preparation and the review was difficult due to time restrictions. Especially for phase two, the CIO rated the identification of decisions overall as good. For the task IT-strategy, the CIO rated the description of decision-related information as difficult. He did not want to consider the definition of the IT-strategy as a decision, although the interview had identified 5 decisions. The CIO's feedback to the artifacts (m4) is summarized in Table 6.

The CIO's perception of artifacts did not change between phase one and two. All decision-specific information was contained in the artifacts. The CIO perceived the *task description* as important, and easy to understand and comment on. However, he con-

sidered the *Categorization of Subtasks* as difficult to understand and comment on. Once the categorization was done, the CIO understood its importance. Initially, the CIO did not see the importance of the *Decision-specific Task Description*, and hence, rated the understanding as difficult but easy to comment on. After the first review of this description, he rated them as important. He perceived the description of complex processes (such as decisions) as challenging. The CIO rated the description of *data sources* as very easy to understand, but difficult to comment on (and fill in), since he was not aware of detail knowledge about URLs. The CIO perceived *Workspaces* as a complex summary of task details and rated them as understandable, but unimportant to him, since they are an intermediate step to virtual windows. The CIO perceived the *UI Structure Diagram* as a complex and overloaded representation and rated them as difficult to understand and comment. He recommends to hide workspace details and focus the navigation between them. The *UI Prototype* gives the most detailed description of the system-to-be and allows to check whether the UI contains the necessary information. It was rated as very easy to understand and comment and as a very important artifact. The CIO rated the developed system prototype (m5) as useful, appropriate, and applicable with a good user friendliness. However, a more detailed system test is yet to be done by the CIO to identify missing data or other gaps. He definitely will employ the prototype. The requirements engineer (m6) rated the detailed understanding of the artifact types and their relations to each other as very important to structure the interviews. The requirements engineer needs to understand the RE process to choose an appropriate sequence of DPs to visit and artifacts to create. During the RE phase, s/he must keep all artifacts consistent, which is challenging. The DsTORE artifacts enabled a detailed understanding of the decisions. The specification was extremely helpful to guide the developers during the implementation of the prototype, in particular the knowledge of the required data and their origin. Therefore RQ.1 can be answered as follows: DsTORE supports the specification of requirements for a PDSS adequately. It allows an early and continuous stakeholder feedback, emphasizes the decision-specific data, and decision-specific UI prototyping.

7 DISCUSSION

In this section, we discuss the results of the DsTORE evaluation and possible improvements to DsTORE, as well as lessons learned and general experiences with the method adaptation.

Discussion of DsTORE: The *problem investigation* shows that TORE as a structured RE framework provides the basis for the specification of a PDSS. This is not surprising as PDSS share some properties with IS (cf. Table 1). The missing decision-specifics are provided by DsTORE. Small adaptations in DsTORE help to understand decisions and to gain a detailed specification of all PDSS aspects. The *treatment validation* shows that DsTORE is accepted by the stakeholder and the requirements engineer, although there is potential for improvement.

The effort of 106 hrs to create the artifacts is very high, but decreased significantly in phase two based on the prior experience. Four interviews with 7 hrs and two interviews with 3 hrs is a reasonable effort. Based on the CIO's feedback, the semi-structured interviews are adequate to elicit requirements and allow a creative discussion. The results show that the CIO views atomic decisions as part of larger business- or management processes which are also important to him. It therefore appears interesting to study the relations of tasks and their atomic decisions to management processes. The CIO rated only the artifact workspaces as less important for him. We used the workspace to structure the virtual windows. In the future, workspaces might be used only by the requirements engineer while the structure is discussed with the stakeholder directly in the virtual windows. The CIO's rating emphasizes the importance of the data sources and the UI. The CIO's suggestions do not concern the DPs, but rather the execution of the interviews, which can easily be improved. E.g. decisions can be identified on the task level with the CIO without specifying details, and can then be detailed with external experts. The latter can also supply the data source details. The categorization of subtasks can be done initially by the requirements engineer and presented to the CIO for review. The artifacts' consistency can be improved by a yet to define tool support. Both, CIO and requirements engineer, missed a RE-process description covering activities and artifacts. This can easily be provided in the future.

Lessons Learned: The following sequence is helpful for creating some of the artifacts of DsTORE. (1) task and subtask (2) detailed decision description (3) domain data model (4) virtual windows (5) workspaces (6) UI Structure Diagram (7) UI Prototype. A template for the naming of subtasks should be used, i.e. <verb | nominalized verbs> <object>. Synonyms for similar verbs should be avoided. During the elicitation of the to-be decision-specific subtask description, it is important to continuously ask the stakeholder which data exactly is necessary for

the decision. It is important to distinguish explicitly between domain entities and attributes, as this is not done directly by the stakeholder. The specification of rules is often awkward, especially when they are complex, or have not been formulated previously. It was helpful for us to assure that attributes of rules can be mapped to the domain data model. It is worth the effort to assemble the results of the decision description into virtual windows as early as possible.

General Experiences with Method Adaptation: During the *treatment design* we developed several further artifact types, which we finally discarded or simplified, so that they could fit into DsTORE artifact types. The design science process helped to structure the adaptation of TORE into problem identification, specific adaptations, and their evaluation. It forced us to create explicit justifications for the adaptations. Further, it forced us to collect explicit feedback from the stakeholders involved in the method execution. It is important to discuss the adaptations with independent experts not involved in the method execution.

8 THREATS TO VALIDITY

The threats to validity are structured according to Runeson et al. (Runeson and Höst, 2009). **Construct validity** considers whether the study measures what it claims. The artifact types of DsTORE were not yet fixed during phase 1. As all DsTORE DPs as presented in Section 5 were applied in treatment validation, we believe that the DsTORE artifact types that have been temporarily used and discarded do not distort the final results. **Internal validity** considers causal relations of investigated factors, i.e. effects of unknown factors that influence an investigated factor. Training obviously affected the effort and the opinion of the stakeholders. The effort decreased between the case studies, owing to training effects of the requirements engineer and CIO. We asked the CIO about his opinion in activity 9, after he had gained the experience. The system test and the implementation of the results of the case study, phase two, might change the evaluation. However, we consider this as unlikely. **External Validity** describes the generalizability of the findings and the transfer of study results outside the study. The case studies are based on a single case only. The case study involved only one person. DsTORE is specific to PDSSs and the transferability to other DSS types is unclear. **Reliability** considers the influence of the specific researcher and indicates threats to validity for a repetition of the study. The main threat to reliability is that the first author (as requirements engineer) had much influence on the development and process of the case study, in particular during the interviews and the data analysis. We mitigated this by continuous discussion with the 2nd researcher.

9 CONCLUSION

In this study, we presented a *problem investigation*, DsTORE as the *treatment design* which extends TORE to support requirements engineering for PDSS, and a *case study* that evaluates DsTORE in the case of a PDSS, in order to show its basic feasibility. The results are encouraging to continue with the following future work: First, we will extend the prototype with the task *IT-strategy*. After the system test with the CIO, this prototype will be put into operation at the hospital. Second, we will consolidate the improvements of DsTORE as we have sketched above. Third, as data play a prominent role, we will explore how an ontology can support the RE process. In carrying out this last aspect, we rely on the SNIK domain ontology (Schaaf et al., 2015) for IM in hospitals. We want to elaborate, how this knowledge can be used in combination with DsTORE to further improve the requirements elicitation and specification. Fourth, we believe that it would be interesting to investigate whether and how the presented approach also supports the specification of other types of DSSs.

ACKNOWLEDGEMENTS

We thank the involved CIO for his time and motivated collaboration and the SNIK project team for their intensive cooperation. This work was supported by DFG (German Research Foundation) under the Project SNIK: Semantic Network of Information Management in Hospitals, Grant no. 1605/7-1 and 1387/8-1.

REFERENCES

Adam, S., Doerr, J., Eisenbarth, M., and Gross, A. (2009). Using Task-oriented Requirements Engineering in Different Domains – Experience of Application in Research and Industry. *Proceedings of 17th IEEE International Requirements Engineering Conference (RE'09)*, pages 267–272.

Ammenwerth, E., Haux, R., Knaup-Gregori, P., and Winter, A. (2015). *IT-Projektmanagement im Gesundheitswesen*. Schattauer, Stuttgart, 2. edition.

Arnott, D. (2008). Personal Decision Support Systems. In Burstein, F. and Holsapple, C., editors, *Handbook of Decision Support Systems 2*, chapter 43, pages 127–150. Springer-Verlag Berlin Heidelberg.

Arnott, D. (2010). Senior executive information behaviors and decision support. *Journal of Decision Systems*, 19(4):465–480.

Arnott, D. and Pervan, G. (2005). A critical analysis of Decision Support Systems research. *Journal of Information Technology*, 20(2):67–87.

Barone, D., Topaloglou, T., and Mylopoulos, J. (2012). Business Intelligence Modeling in Action: A Hospital Case Study. In Jolita Ralyte, Franch, X., Brinkkemper, S., and Wrycza, S., editors, *Advanced information systems engineering*, pages 502–517.

Gachet, A. and Haettenschwiler, P. (2006). Development processes of intelligent decision-making support systems: review and perspective. In *Intelligent Decision-making Support Systems*, pages 97–121. Springer.

Gao, S. (2013). Mobile decision support systems research: a literature analysis. *Journal of Decision Systems*, 22(1):10–27.

García, S., Romero, O., and Raventós, R. (2016). DSS from an RE Perspective: A systematic mapping. *Journal of Systems and Software*, 117:488–507.

Giorgini, P., Rizzi, S., and Garzetti, M. (2008). GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems*, 45(1):4–21.

Holsapple, C. W. (2008a). Decisions and Knowledge. In Burstein, F., editor, *Handbook on Decision Support Systems 1*, chapter 2, pages 21–53. Springer.

Holsapple, C. W. (2008b). DSS Architecture and Types. In Burstein, F., editor, *Handbook on Decision Support Systems 1*, chapter 9, page 854. Springer.

Hosack, B., Hall, D., Paradice, D., and Courtney, J. (2012). A Look Toward the Future: Decision Support Systems Research is Alive and Well. *Journal of the Association for Information*, 13(Special Issue):315–340.

Jahn, F., Schaaf, M., Paech, B., and Winter, A. (2014). Ein semantisches Netz des Informationsmanagements im Krankenhaus. In *Informatik 2014*, volume LNI P-232, pages 1491–1498.

Kücherer, C., Jung, M., Jahn, F., Schaaf, M., Tahar, K., Paech, B., and Winter, A. (2015). System Analysis of Information Management. In *Informatik 2015*, volume LNI P-246, pages 783–796, Cottbus, Germany.

Lauesen, S. (2005). *User interface design*. Pearson/Addison-Wesley, Harlow;Munich.

Paech, B. and Kohler, K. (2004). Task-driven requirements in object-oriented development. In J. D. Leite, J., editor, *Doorn, Jorge H. Perspectives on Software Requirements.*, volume 753, pages 1–25. Kluwer Academic, 2004. Print.

Pourshahid, A., Johari, I., Richards, G., Amyot, D., and Akhigbe, O. S. (2014). A goal-oriented , business intelligence-supported decision-making methodology. *Decision Analytics Journal*, 9(1):1–36.

Power, D. J. (2011). What are the features of a document-driven dss? Technical report, DSS News. Accessed: 2016-08-08.

Runeson, P. and Höst, M. (2009). Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Softw. Eng.*, 14(2):131–164.

Salinesi, C. and Gam, I. (2009). How specific should Requirements Engineering be in the context of Decision Information Systems? In *RCIS 2009: Proceedings of the Third International Conference on Research Challenges in Information Science*, pages 247–254, Fez, Morocco. IEEE.

Saxena, K. B. C. (1991). Decision support engineering: a DSS development methodology. In *Proceedings of the 24th Annual Hawaii International Conference on System Sciences (HICSS '91)*, volume 3, pages 98–107.

Schaaf, M., Jahn, F., Tahar, K., Kücherer, C., Winter, A., and Paech, B. (2015). Entwicklung und Einsatz einer Domänenontologie des Informationsmanagements im Krankenhaus. In *Informatik 2015*, volume LNI P-246 of *Lecture Notes in Informatics*, Cottbus, Germany.

Topaloglou, T. and Barone, D. (2015). Lessons from a Hospital Business Intelligence Implementation. *Proceeding from CAiSE 2015*, pages 19–33.

Van Solingen, R., Basili, V., Caldiera, G., and Rombach, H. D. (2002). Goal question metric (gqm) approach. *Encyclopedia of software engineering*.

Wieringa, R. J. (2014). *Design Science Methodology for Information Systems and Software Engineering*. Springer Berlin Heidelberg, Heidelberg, Germany; New York, USA; Dordrecht, The Netherlands; London, UK.