The Value of Design Rationale Information

DAVIDE FALESSI
University of Rome Tor Vergata, Italy and Simula Research Laboratory, Norway

LIONEL C. BRIAND
University of Luxembourg, Luxembourg

GIOVANNI CANTONE
University of Rome Tor Vergata, Italy

RAFAEL CAPILLA
Rey Juan Carlos University, Spain

and

PHILIPPE KRUCHTEN
University of British Columbia, Canada

A complete and detailed (full) Design Rationale Documentation (DRD) could support many software development activities, such as an impact analysis or a major redesign. However, this is typically too onerous for systematic industrial use as it is not cost-effective to write, maintain, or read. The key idea investigated in this paper is that DRD should be developed only to the extent required to support activities particularly difficult to execute or in need of significant improvement in a particular context. The aim of this paper is to empirically investigate the customization of the DRD by documenting only the information items that will probably be required for executing an activity. This customization strategy relies on the hypothesis that the value of a specific DRD information item depends on its category (e.g., Assumptions, Related requirements, etc.) and on the activity it is meant to support. We investigate this hypothesis through two controlled experiments involving a total of seventy-five master students as experimental subjects. Results show that the value of a DRD information item significantly depends on its category and, within a given category, on the activity it supports. Furthermore, on average among activities, documenting only the information items that have been required at least half of the time (i.e., the information that will probably be required in the future) leads to a customized DRD containing about half the information items of a full documentation. We expect that such a significant reduction in DRD information should mitigate the effects of some inhibitors that currently prevent practitioners from documenting design decision rationale.

Categories and Subject Descriptors: D.2.10 [Software-Design]: Representation

Additional Key Words and Phrases: Empirical software engineering, software architecture, design decisions, value-based software engineering, software maintenance.

1. INTRODUCTION

During any software development process, most architectural design decisions are not explicitly documented with their rationale, as they are often embedded in the models the architects build [Tyree and Akerman 2005]. Consequently, useful knowledge associated to the decision-making activities is lost forever [Falessi, et al. 2011, Kruchten, et al. 2009].

This work of Lionel Briand and Davide Falessi was supported by a grant from the Norwegian research council. This paper is a revised and extended version of a paper presented at WICSA 2008 (Vancouver, Canada) and SHARK 2008 (Leipzig, Germany).

Author’s addresses: D. Falessi, University of Rome – Tor Vergata, DISP and Simula Research Laboratory, Certus Software V&V Center; L. C. Briand, SnT Centre for Security, Reliability, and Trust, University of Luxembourg; G. Cantone, University of Rome – Tor Vergata, DISP; R. Capilla, Rey Juan Carlos University, DCS; P. Kruchten, University of British Columbia, ECE. Permission to make digital/hard copy of part of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date of appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.
In cases where the design erodes, the problem of knowledge vaporization [Bosch 2004], often due to a lack of Design Rationale Documentation (DRD), leads to high maintenance cost, as new design decisions cannot rely on previous ones.

Although the use of design rationale is recognized as one of the most promising steps for advancing the state of the art of software architecture design and maintenance [Bosch 2004], its widespread industrial use has been hindered by socio-technical inhibitors, in particular the effort to produce and maintain additional documentation [Lee 1997]. According to Gorton [Gorton 2006], “Generating architecture documentation is nearly always a good idea. The trick is to spend just enough effort to produce only documentation that will be useful for the project’s various stakeholders. This takes some upfront planning and thinking.” In other words, DRD should be tailored to support activities that are the most in need of improvement in a given context.

The term information item refers to a single piece of rationale information regarding a design decision. The taxonomy of DRD information proposed by Tyree and Akerman [Tyree and Akerman 2005] provides a categorization of information items for a design decision. According to this categorization, an information item always belongs to a single category.

The key idea we have investigated is that design rationale documentation should only be introduced to the extent required to support subsequent activities that are particularly challenging to perform, or in significant need of improvement in a particular context. Hence, the purpose of this paper is to report on an empirical study of the customization of the design rationale documentation by restricting the documentation to only the information elements that are very likely to be required to perform a subsequent activity.

In order to demonstrate the feasibility and value of DRD customization we investigate here the following research questions:

**R.Q. 1:** Is the value of an information item significantly affected by its category and the activity it supports?

**R.Q. 2:** How much effort could be saved by adopting a value-based DRD?

The empirical procedure consists of two controlled experiments performed in two different geographical locations and both involving trained, graduate students. The current paper combines the data of the first experiment with that of its replica in order to gain in statistical power and apply more sophisticated analysis techniques. More specifically, we have used Multiple Correspondence Analysis [Greenacre 2007]—a technique dedicated to large contingency tables (categorical data) and their
interpretation—to analyze the relations among the category of DRD information items, its value, and the specific activities it supports.

We decided to focus on DRD and do not consider other types of existing documentation because there is a general agreement that sharing architectural knowledge is a very relevant problem and it is not addressed by any documentation other than DRD [Lago, et al. 2008].

The remainder of this paper is structured as follows: Section 2 presents the related work and introduces the concepts used in this study. Section 3 discusses the costs and benefits of using design rationale and presents our key idea. Section 4 describes two experiments; a controlled experiment and an exact replica, with the goal to assess empirically the feasibility and efficiency of documenting only the information items that are valuable for the intended use or purpose. Section 5 reports the empirical results and their discussion. Section 6 discusses in great details how the work presented in this paper can be built upon to eventually support the application of DRD. The paper concludes in Section 7.

2. RELATED WORK

2.1 DRD Approaches and Tools

There are many definitions of design rationale [de Boer and Farenhorst 2008]; one of the most comprehensive definitions has been proposed by Jintae Lee: “Design rationales include not only the reasons behind a design decision but also the justification for it, the other alternatives considered, the tradeoffs evaluated, and the argumentation that led to the decision.” [Lee 1997] The rationale can be classified into several types; most of the times these types are not mutually exclusive. Burge and Brown in [Burge and Brown 1998] propose the following types of rationale: argumentation, history, device, process, and active document. In particular, in the argumentation-based DRD, design rationale is used to represent the arguments that characterize a design, such as, issues raised, alternative responses to these issues, and arguments for and against each alternative. Prominent argument-based design rationale techniques are gIBIS [Conklin and Begeman 1988], DRL [Lee 1997], and QOC [MacLean, et al. 1996].

Lee states that much of the design rationale is embedded in design specifications or parts of meeting discussions [Lee 1997]. Addressing all the issues that match the different dimensions of design rationale is also difficult because different stakeholders are interested in different concerns. This interest in different concerns is similar in principle.
to our value-based DRD approach which is based on the premise that “what you represent depends on what you want to do with it” [Lee 1997].

Tang et al. evaluated the importance of DRD as perceived by industrial software architects [Tang, et al. 2007]. Our paper shares with Tang et al. the view that “practitioners recognize the importance of documenting design rationale […] however they have indicated barriers to the use and documentation of design rationale” [Tang, et al. 2007]. In particular, they investigated which type of information is generally more likely to be used by practitioners. We have tried to go a step further by investigating the level of support, provided by each category of information, for specific activities.

Referring to the knowledge capturing problem, Tyree and Akerman proposed a framework to document design decision rationale for system architectures [Tyree and Akerman 2005]. In the present study, we use such a documentation template as an example of DRD and we investigate the level of usefulness of each information item category in this template in support of different software development activities.

Capilla et al. described an approach for modeling and documenting the evolution of architectural design decisions that is characterized by sets of mandatory and optional attributes that can be tailored according to different users’ needs as well as to different organizations [Capilla, et al. 2007]. Customized information is used to adapt part of the information captured for the design decisions to the specific needs of different stakeholders and organizations.

Kruchten et al. had suggested a set of different activities supported by DRD [Kruchten, et al. 2006]; we investigate some of these (e.g., impact analysis) in the present paper.

Van der Ven et al. describe that different types of stakeholders adopt DRD to enact specific activities; in particular they presented a use-case model that arose from industrial needs [Van der Ven, et al. 2006].

Farenhorst et al. recently investigated by means of a large-scale survey the behavior of architects in terms of their daily activities, and how important they consider the various types of support for sharing architectural knowledge [Farenhorst, et al. 2009]. Their results indicate that architects mainly consume architectural knowledge, but neglect to document and share such knowledge themselves. Such a result calls for supporting architectural knowledge sharing by effectively balancing the costs and benefits of design rationale information.

Jansen et al. presented the Architectural Design Decision Recovery Approach (ADDRA) for recovering architectural design decisions after the fact [Jansen and Bosch
2005]. In particular, ADDRA uses architectural deltas to provide the architect with clues about these design decisions. This DRD approach has the advantage of requiring little effort from the DRD producer.

Lee and Kruchten divided the documentation activity into three steps: flagging information (identification of possible significant information related to a decision that is being made), filtering (excluding some of the information selected in the previous step), and forming (merging the information produced in the previous step to create a useful and comprehensive DRD) [Lee and Kruchten 2007]. Their key idea is that only the flagging step needs to be enacted near the decision-making process; so most of the effort required by the DRD can be postponed according to the decision-maker availability.

2.2 Value-based Approaches

We consider all the above-mentioned DRD approaches as value-neutral because they do not relate to a particular business context; rather they aim to maximize the benefits for the knowledge consumer, by imposing the burden on the knowledge producer to document all potential useful information. To date, “much of current software engineering practice and research is done in a value-neutral setting, in which every requirement, use case, object, test case, and defect is equally important” [Biffl, et al. 2006]. Consequently, “a resulting value-based software engineering agenda has emerged, with the objective of integrating value considerations into the full range of existing and emerging software engineering principles and practices, and of developing an overall framework in which they compatibly reinforce each other” [Biffl, et al. 2006]. In the present work, we apply value-based software engineering principles to documentation, and we propose a value-based approach to DRD consisting in prioritizing the information to document according to the activity to support, i.e., the activities particularly hard to enact without DRD.

The idea of applying a value-based approach to requirements traces was proposed in several studies including [Arkley, et al. 2006, Egyed, et al. 2005, Egyed, et al. 2007, Falessi, et al. 2011, Heindl and Biffl 2005]. We share with these studies the aim and vision though the object being tailored is different (requirements trace vs. DRD). Moreover, “How much traceability is enough?” [Cleland-Huang 2006] and “When and How does Requirements Traceability Deliver More Than it Costs?” [Cleland-Huang 2006], were addressed in panel discussions held at two important international conferences: COMPSAC06 and RE06.

More than twenty years ago, Basili and Rombach provided practical guidelines for a successful reuse strategy [Basili and Rombach 1991]. Similarly, in our work we propose
a guideline to reuse architectural knowledge: the model capturing experience (i.e., DRD) should be tailored to specific project objectives in terms of activities to support.

The present work perfectly matches the agile modeling approach [Falessi, et al. 2010], and in particular “The TAGRI Principle of Software Development: They Ain't Gonna Read It”, as a way to cut the documentation effort [Ambler and Jeffries 2002]. While Ambler poses some relevant questions regarding the different ways stakeholders use the documentation [Ambler 2007], in this paper we provide quantitative results regarding the extent to which specific rationale information items support specific activities.

2.3 DRD Benefits
The role of design rationale in software engineering has been extensively discussed by Burge et al. [Burge, et al. 2008]. DRD can support several software architecture related activities: architectural review, review for a specific concern, change impact analysis, studying the chronology, adding decisions, cleaning up the system, spotting the subversive stakeholder, cloning architectural knowledge, detection and interpretation of patterns [Kruchten, et al. 2006]. To explain the meaning of such DRD uses, let us consider, for instance, the activity ‘Impact analysis’, which is concerned with the management of requirement changes. In practice, changes in requirements or business goals are very frequent. Martin Fowler [Fowler 2005] emphasized the unpredictability of requirements: “What might be a good set of requirements now, is not a good set in six months time. Even if customers can fix their requirements, the business world isn't going to stop for them.” In the case of a requirements change, it is crucial to understand which decisions (and eventually which system artifacts) are still valid and which ones have to be re-worked (i.e., redesigned and/or re-implemented).

Several researchers assessed the support provided by DRD. For instance, Karsenty assessed the QOC approach in the maintenance of a nine-month old software project [Karsenty 1996].

Brathall et al. presented a controlled experiment to evaluate the importance of DRD when predicting change impact on software architecture evolution [Brathall, et al. 2000]. Results show that DRD clearly improves effectiveness and efficiency.

Zimmermann et al. presented a proactive approach to modeling and reusing architectural knowledge for enterprise application development [Zimmermann, et al. 2007]. Their approach has already shown to be practical for BPM requirement models.
and the SOA architectural style: they observed initial effort savings and quality improvements on an early adoption project.

Falessi et al. analyzed the value of DRD with respect to effectiveness and efficiency of individual/team decision making in the presence of requirement changes. The main goal was to estimate whether the availability of the DRD would improve the correctness of design decisions. The study was a controlled experiment in which fifty Master students were the experiment subjects. Figure 1 summarizes the results of the study; we can see how in case of requirement changes the correctness of the decisions improves when the DRD is available for decision makers, both for individual participants and teams. Given the empirical evidence that DRD is beneficial, the natural next step is to provide realistic means to reduce the inhibitors for DRD usage.

![Figure 1: Correctness of individual decisions and team decisions with and without Design Rationale Documentation.](image)

3. BALANCING THE COSTS AND BENEFITS OF DESIGN RATIONALE INFORMATION

3.1 DRD Inhibitors

Although several studies empirically demonstrated that the use of design rationale documentation brings numerous benefits [Bratthall, et al. 2000, Falessi, et al. 2006, Karsenty 1996], such type of documentation is not widely adopted in practice. Through discussions with our industrial partners we’ve identified the following inhibitors to the adoption of DRD:

- **Bad timing and delayed benefit.** The period in which design decisions are made is often critical for the overall project success. People involved in design decisions are usually busy trying to perform other more recognized and essential tasks and to meet their related deadlines. In such circumstances, documenting
rationale is perceived to be less important and is eventually dismissed. Our experience shows that when we suggest documenting the design decision rationale in an appropriate time frame, the most common answer is: “We are already under pressure to meet the deadline, investing additional time in documentation would make the situation worse”.

- **Information predictability.** DRD consumers and producers are often different persons. People who are responsible for evolving a software project are usually not the original designers, who may not even be part of the project organization anymore. Hence, the documentation producer needs to forecast which information the consumers will need in the future. As a result, the producer would need to document all the information that could be useful.

- **Overhead.** Several DRD techniques already exist [Burge and Brown 1998, Conklin and Begeman 1988, Jansen, et al. 2008, Lee and Kruchten 2007]; however, they usually focus on maximizing the consumer benefits rather than minimizing the producer effort. This results in substantial effort being spent on documentation and maintenance activities. Supposedly, the overhead required to capture design rationale information is regained by assisting future maintenance and evolution activities. This overhead can be minimized by carefully selecting the DRD information items to be captured. Shum and Hammond in [Shum and Hammond 1994] pointed out that without a good Return On Investment (ROI), the documentation management system would not be used or would ultimately be counterproductive.

- **Unclear benefits.** Decision-makers often do not know how the DRD will support specific activities.

- **Lack of motivation.** This may arise from the absence of direct benefits or a lack of personal interest. People in charge of documenting and maintaining DRD artifacts (the decision makers) are not very motivated because they do not directly benefit from DRD. Lee in [Lee 1997] had already raised the issue of having distinct DRD producers and consumers. Moreover, experts may not be interested in making their valuable knowledge explicit as they may perceive it to be an asset. In other words, some experts may see no clear advantage in documenting design rationale.

- **Lack of maturity.** Only few tools are currently available to support DRD and the majority of them are still immature [Burge and Brown 1998, Conklin and Begeman 1988, Jansen, et al. 2008, Lee and Kruchten 2007].
• **Potential inconsistencies.** DRD and designs should be kept up to date and aligned to avoid potential inconsistencies when the design is modified or when decisions change.

3.2 Key Idea: a Value-Based Customization

Though all the inhibitors exist, the “Overhead” seems the most important to address because it would in turn impact additional inhibitors including “Bad timing and delayed benefits”, “Lack of motivation”, and “Potential inconsistencies”. For this reason, we decided to focus here on the “Overhead” inhibitor. The key idea is to customize the DRD by documenting only the information items that will probably be required for executing a particular software architecture-related activity [Kruchten, et al. 2006], e.g., detecting requirements misunderstandings.

We consider the design rationale documentation as a set of information items. Figure 2 describes the values of different information items; i.e., the level of support to a specific activity related to software architecture. The value of a DRD information item can be:

- **Useless:** The information item does not provide any support to the activity.
- **Optional:** The information item facilitates the activity but is not required.
- **Required:** The information item is required for performing the activity.

![Figure 2: The possible values of an information item.](image)

In the past, research goals focused on maximizing the DRD consumer’s benefit by forcing the producer to document all the information potentially useful for all activities. The DRD is about making the decision-making process reusable. In the software reuse area, it is agreed that focusing the investment on the most valuable asset to reuse is a key factor for a successful reuse strategy [Clements, et al. 2005, Favaro, et al. 1998, John, et al. 2006]. Therefore, it is unrealistic to target all possible elements for reuse, for example documenting all the information items of a DRD. While in past studies we have evaluated the cost and benefits of DRD [Capilla, et al. 2008, Falessi, et al. 2006], in the present
study we empirically evaluate the feasibility of a tradeoff between its costs and benefits. The key idea is to compromise between the cost to the producer and the benefit to the consumer by achieving a value-based customization (see Figure 3), which consists in documenting only the information items that are likely to be valuable for the intended purpose.

We do not imagine a tailored DRD to be a panacea. One disadvantage is that the activities DRD is meant to support have to be known when architects make design decisions. Even though this is an important issue, given the presence of several inhibitors, it is reasonable to consider DRD as an economic investment; thus, DRD needs to be motivated by the presence of a real business case. Therefore, DRD should be introduced to support activities that are particularly difficult to enact with the usual procedures and documentation.

We don’t propose a specific process to document design rationale; we propose to document only the information items that are expected to be valuable for selected activities.

4. EXPERIMENT PLANNING

Given the abovementioned benefits and inhibitors regarding DRD adoption, we now present two controlled experiments investigating the feasibility and efficiency of documenting only the DRD information items that are valuable for the intended use or purpose. This section is structured, in compliance with [Jedlitschka, et al. 2008], as
follows. We firstly present the objectives of the empirical investigation. Then we describe the rationale of the applied empirical methodology. Next, we describe the subjects and the variables that characterize the experiments. We then proceed by reporting on the experimental tasks and present the adopted material and design. We conclude by discussing the procedures we applied to prepare the experiment and the approach we followed to collect and validate the data.

4.1 Research Questions and Hypotheses

Our research questions are:

- **R.Q. 1:** Is the value of an information item significantly affected by its category and the activity that it supports?

  The software architecture field is not mature from an empirical perspective [24, 28] as there is no available objective measure for the value of a DRD information item. Hence, our strategy is to measure the frequency by which it is required by subjects for performing a given activity. A value-based customization of design rationale documentation relies on the following hypothesis: the value of an information item is affected by its category and the activity it aims to support. If that is true, architects can reduce the number of information items to document by selecting the ones expected to support a given activity of interest.

- **R.Q. 2:** How much effort can be saved by adopting a value-based DRD?

  Once the feasibility of a value-based approach is established, then it is important to assess its efficiency in terms of saved documentation effort. Reducing the effort required to document design rationale is essential to overcome DRD inhibitors. In this study a piece of DRD information is deemed useful to document if most subjects perceive it is required for supporting a given activity. Thus the reduced effort is estimated by analyzing the difference between the number of information items of a full DRD versus a customized DRD. See Section 5.3.2 for a detailed discussion about this measurement procedure.

4.2 Research methodology

The research methodology consists of a controlled experiment and its exact replica. In general, one classical method for identifying cause-effect relationships is to conduct controlled experiments where only independent variables vary [56] and other factors are either controlled or their effects mitigated. Our decision to adopt an experiment stems from the many uncontrolled, confounding factors that could blur the results in an industrial context. Thus, in order to assess the feasibility and potential benefits of a value-
based DRD we enacted a controlled experiment involving master students in computer and electrical engineering at the University of Rome Tor Vergata (Italy).

In general, the purpose of replication is to help the research community build knowledge about which results hold under which conditions [Basili, et al. 1999]. In an exact replica, changes to the original experiment concern only the context in which the study is conducted (e.g., population, languages, etc.) and not the procedure that the subjects follow [Shull, et al. 2008]. Its main benefit is to help mature the software engineering body of knowledge by addressing both conclusion and internal validity problems in existing experiments. Regarding internal validity, an exact replica aims to confirm that the variables taken into account (i.e., the replicated ones) are the ones influencing results [Shull, et al. 2008]. In other words, an exact replica providing a result different from the original experiment would suggest the existence of unknown, confounding variables. Regarding conclusion validity, an exact replica increases statistical power by providing further data to analyze, together with the original experiment.

Therefore, in order to increase our confidence in the results, we ran an exact replica [Shull, et al. 2008] of our controlled experiment at the Rey Juan Carlos University of Madrid (URJC) in Spain.

Though we did not change the experimental settings, there are some minor differences among the experiment and its replica: the subjects’ background was computer engineering, and computer science in the experiment and replica, respectively. Moreover the experimental material was translated from Italian to Spanish.

4.3 Experimental units

Fifty graduate students belonging to a Master’s course in computer engineering at the University of Rome TorVergata (Rome) participated in the experiment. Twenty-five graduate students belonging to a Master’s course in computer science at the Rey Juan Carlos University (URJC) participated in the replica.

All the students attended extensive teaching on the various phases of the software lifecycle through their bachelor and master courses, including requirements engineering and software architecture analysis and design. Some of the students had industrial experience or worked as private consultants.

Because the use of students as subjects can be considered as one of the main threats to validity of the present study, we provide a detailed discussion of this issue in Section 5.3.4.
4.4 Variables

The output and dependent variable is the Value of an information item in a DRD. Value represents the level of support it is perceived to provide by the subjects for performing an activity. We measure the Value on a 3-point ordinal scale (Useless, Optional, or Required), for each specific information item, after having executed an activity. This measurement mechanism is based on the subjective theory of value [Cox 1997]: the value is based on the needs of subjects using an object (i.e., DRD) rather than on the inherent importance of to the object itself. An alternative measurement mechanism was taken in [Falessi, et al. 2006] where the impact of DRD was measured by its influence on decision correctness. Since multiple DRD categories have a confounded effect on decision correctness, the theory of subjective value is the only viable option to discriminate among the value of DRD information categories.

Among the independent variables, we considered the following two factors:

- The activity performed by subjects (Activity). Since we had: (1) fifty and twenty-five subjects in the original experiment and replica, respectively, (2) up to five areas of expertise (as detailed in Section 4.7), and (3) the need to make the design as balanced as possible, we decided to use five activities. Among all possible relevant activities [Kruchten, et al. 2006], we selected the five activities based on the level of expected validity threats given the available experiment time and the subjects’ experience. Such an expectation was based on our experience in conducting similar experiments in controlled environments and with master students [Capilla, et al. 2008, Falessi and Cantone 2006, Falessi, et al. 2006, Falessi, et al. 2007]. Specifically we analyzed the following five activities:
  - Detecting wrong aspects in decisions. Which characteristics of the chosen decision are wrong?
  - Detecting requirements misunderstandings. Which characteristics of the problem to address have been misunderstood?
  - Checking design: verification and evaluation. Do you approve the decision made?
  - Detecting conflicts between new requirements and an old decision. Is the old decision still valid for the new set of requirements?
  - Evaluating impact. What is the impact of new requirements on the system?
- The category of the information item (Category). In the present study we assigned 13 levels to this factor (Table 1) as proposed by Tyree and Akerman in [Tyree and Akerman 2005] for documenting design decisions.
Table 1: Categories of information of design decision rationale documentation as proposed in [Tyree and Akerman 2005].

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue</td>
<td>Describe the architectural design issue you’re addressing, leaving no questions about why you’re addressing this issue now. Following a minimalist approach, address and document only the issues that need addressing at various points in the life cycle.</td>
</tr>
<tr>
<td>Decision</td>
<td>Clearly state the architecture’s direction—that is, the position you’ve selected.</td>
</tr>
<tr>
<td>Status</td>
<td>The decision’s status, such as pending, decided, or approved.</td>
</tr>
<tr>
<td>Assumptions</td>
<td>Clearly describe the underlying assumptions in the environment in which you’re making the decision—cost, schedule, technology, and so on. Note that environmental constraints might limit the alternatives you consider.</td>
</tr>
<tr>
<td>Constraints</td>
<td>Capture any additional constraints to the environment that the chosen alternative might pose.</td>
</tr>
<tr>
<td>Positions</td>
<td>List the positions (viable options or alternatives) you considered. These often require long explanations, sometimes even models and diagrams. This isn’t an exhaustive list. This section also helps ensure that you heard others’ opinions; explicitly stating other opinions helps enroll their advocates in your decision.</td>
</tr>
<tr>
<td>Argument</td>
<td>Outline why you selected a position, including items such as implementation cost, total ownership cost, time to market, and required development resources’ availability.</td>
</tr>
<tr>
<td>Implications</td>
<td>A decision comes with many implications. Clearly understanding and stating your decision’s implications can be very effective in gaining buy-in and creating a roadmap for architecture execution.</td>
</tr>
<tr>
<td>Related decisions</td>
<td>It’s obvious that many decisions are related; you can list them here. However, we’ve found that in practice, a traceability matrix, decision trees, or metamodels are more useful. Metamodels are useful for showing complex relationships diagrammatically.</td>
</tr>
<tr>
<td>Related requirements</td>
<td>Decisions should be business driven. To show accountability, explicitly map your decisions to the objectives or requirements. You can enumerate these related requirements here, but we’ve found it more convenient to reference a traceability matrix. You can assess each architecture decision’s contribution to meeting each requirement, and then assess how well the requirement is met across all decisions.</td>
</tr>
<tr>
<td>Related artifacts</td>
<td>List the related architecture, design, or scope documents that this decision impacts.</td>
</tr>
<tr>
<td>Related principles</td>
<td>If the enterprise has an agreed-upon set of principles, make sure the decision is consistent with one or more of them. This helps ensure alignment along domains or systems.</td>
</tr>
<tr>
<td>Notes</td>
<td>Because the decision-making process can take weeks, we’ve found it useful to capture notes and issues that the team discusses during the socialization process.</td>
</tr>
</tbody>
</table>

4.5 Tasks

This work concerns the study of the impact of DRD on architecture relevant decisions. Therefore, decisions are the units of analysis and the experimental tasks are activities involving such decisions based on DRD. Specifically, during the experiment, subjects received the description of a system (e.g., similar to a vision document in RUP [Kruchten 2003]) and a set of five decisions and their DRD. Each subject performed one activity on each decision in the experiment and five activities on each decision in the replica. See the list of activities in Section 4.4 for further details. For each of the five decisions, the subjects had to execute the following steps:

- Understand the activity.
- Note the starting time.
- Read and understand the DRD related to a specific decision.
- Execute the activity and write the requested answer. Note that, though the answer’s correctness is not relevant for our research questions, we need to ensure that subjects completed the activities assigned to them.
• Note the ending time. Note that this data is not needed for addressing our research questions but only for data quality assurance.
• For each DRD category, score how much support DRD provided to the activity: Useless, Optional, or Required.

Steps 2 and 5 are needed to check the correctness of the order of execution of the activities, as we detail in Section 4.10.

4.6 Experimental Material

The experimental material was created by the authors of this paper with the aim to achieve realism. However, we chose to use artificial design elements instead of real ones given the time constraints of a controlled environment and the need to focus on mechanisms (i.e., DRD) and phenomena (i.e., decision-making) of interest. Specifically, we focused on the “decision-making” activity and DRD artifacts, while neglecting other, possibly relevant, activities and types of artifacts. In our context, this meant creating decisions, and related DRD, that would enable realistic decision-making activities without needing further types of documentation and without involving further activities.

The role of artificiality in software engineering experiment is carefully analyzed in [Hannay and Jørgensen 2008]. Their results show that in our field artificiality is usually considered to be mainly a weakness for immediate generalizability of results but they point out that artificiality can be deliberately used for several vital purposes. For instance, as in our case, artificiality helps to achieve control and feasibility in empirical investigations which assess whether a phenomenon is worth further studying.

Our experience in managing real-life architectural decisions guided us towards the production of realistic material. Specifically, to emulate a realistic situation, we devised an artificial software system design which is similar to another, realistic system successfully used in another experimental study [Falessi, et al. 2006]. The project is about a public transportation system with ambient intelligent characteristics [Nehmer, et al. 2006], such as heterogeneous sensors. Moreover, we assume that the software system is at the starting point of its second iteration (of some iterative development process, e.g. RUP [Kruchten 2003]); in the mean time, system requirements did change, and designers, who made the initial design decisions are no longer available to explain such design decisions to the new designers. The system and the high level requirements are described via a document in Word® similar to a vision type of document in RUP. All the material we used was in native languages, that are Italian and Spanish in the original experiment and replica, respectively.
Concerning the requirements changes driving the activities, we selected common change causes such as: (1) variations in industrial strategic partnerships, (2) changes in customer requests resulting from experience using the previous version of the product, and (3) technology advances. One key point is that performing the selected activities (Section 4.4) requires to re-reason about the decisions made during the first iteration of the project.

We adopted a total of twenty-five decisions in order to mitigate the influence of any individual decision on the empirical results. These twenty-five decisions are all, in some sense, architectural decisions; for example:

- The selection of a communication protocol depends on the topology of the nodes, the specific communication mechanism (e.g., publish-subscribe or event-driven) and architectural style (e.g., blackboard or client-server).
- The selection of a data storage mechanism depends on the type of DBMS, the communication protocol, and the architectural pattern (e.g., MVC).

Further drivers for architectural decisions included: available budget, desired compatibility, maintainability, scalability, and security. Constraints, requirements (new and old), rationale, and related decisions (and their status) are described in the provided DRD.

Table 2 shows the form that subjects filled out during the experiment for a given decision. The first column describes the activity to execute. The following three columns describe the initial time, the output of the activity (Answer, column 3), and the final time when the activity was completed, respectively. Columns 5 to 17 describe thirteen levels of Value: columns identify the Category; rows identify the Activity. Thus, each cell in the columns 5 to 17 describes the Value, as perceived by subjects (Useless, Optional, or Required), of a given Category (column), for supporting a given Activity (row).
Table 2: Form that subjects filled in during the experiment. Each cell in the columns 5 to 17 describes the value, as perceived by subjects (Useless, Optional, or Required), of a given DRD category (column), for supporting a given activity (row).

<table>
<thead>
<tr>
<th>Order</th>
<th>Activity</th>
<th>Initial Time</th>
<th>Answer Value of DRD information</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.7 Design

We designed the experiment to make the best possible use of available subjects. In particular, using an artificial project related to ambient intelligence allowed us to model five different areas of expertise: authentication, human interface, operating system, communication protocol, and data storage. Then, subjects expressed their preferences for each area, according to their previous experience and level of confidence. Afterwards, we assigned one area of expertise to each subject and the same number of subjects per area, by satisfying subjects’ preferences to the maximum extent possible. This procedure aimed at ensuring that subjects had sufficient ability and level of confidence in the task to perform. We then prepared five design decisions for each area of expertise. Therefore, each subject dealt with the five decisions pertaining to the assigned area of expertise.

Finally, we designed the experiment to balance:

- Number of activities: All treatments (i.e., the activities described in Section 4.4) have been applied the same number of times on all decisions. This mitigates the influence of specific activities and decisions on the empirical results.
- Order of activities: All activities have been applied the same number of times in all possible orders (i.e., first to fifth). This mitigates the influence of the order in which the treatments are applied on the results.
- There is only one small difference between the original experiment and its replica. In the original experiment the subjects performed one activity per decision on five
decisions. In the replica, to obtain more observations, subjects performed all five activities on each of the five decisions.

4.8 Execution Preparation
The training phase was performed in three sessions of a total of five hours. During the training we taught the concepts of design decisions and design rationale, and the importance of capturing such knowledge. Then, we described the basis and the steps of the controlled experiment and how it would be carried out. We clearly explained almost all the experiment characteristics though we hid the experimenters’ expectations.

We carefully checked for the attendance of the subjects to all the training sessions. As a result, some students were excluded because they did not attend one or more training sessions. In the end, we had fifty-one and twenty-seven students that could be considered properly trained for the experiment and the replica, respectively. Because the experiment was designed to be balanced with twenty-five and fifty subjects, then, to compensate for possible subject absence in the experiment, we randomly selected one and two students as spares for the experiment and replica, respectively.

4.9 Execution Deviations
While conducting the experiment and the replica, no particular deviation from their plan was observed. The experiment was designed to have a suitable number of activity executions given the realistic time a master student is willing to spend in a controlled environment. Subjects had no time constraints. The original experiment and its replica lasted about 90 and 150 minutes, respectively; this difference is explained by the higher number of activities performed by subjects in the replica.

Regarding the experiment, on one occasion where a subject was absent, we replaced him with one of the spare subjects, thus preserving the experiment balance.

4.10 Data Set Collection and Validation
In the original experiment, 50 subjects performed five activities (1 activity per decision) thus yielding 250 observations. Because for each performed activity we had a set of 16 answers to collect (initial time, answer, the Value of each of 13 information categories, and final time), the experiment produced around 4,000 data items, 3,250 of which represents values of DRD information items (13 information items on 250 activity executions). Similarly because in the replica we had 25 subjects enacting 25 activities (5 activities per decision), the replica produced 10,000 data items, 8,125 of which
representing values of DRD information items. The total number of data items is therefore 14,000; 11,375 of which represent values related to DRD information items.

To detect and correct mistakes during the data transcription, we checked the data three times. Moreover, we applied some sanity checks based on simple automatic techniques. For example, for each form, the “initial time” for enacting an activity is checked to: i) precede the “final time” of the same activity, and ii) follow the “final time” of the previous activity. As a result we excluded few invalid data points from any further analysis.

5. RESULTS AND INTERPRETATION

5.1 R.Q. 1: Is the Value of an information item significantly affected by its category and the activity that it supports?

5.1.1 Analysis Procedure

Remember that the original experiment and its replica differ only in the number of activities executions, i.e., five versus twenty-five. Because the training, the design, the activity, the decisions, and the type of subjects were the same in the original experiment and its replica, we decided to merge their data to maximize statistical power. However, we also investigated the consistency of their results as discussed in last paragraph of this subsection.

We visualize the data through histograms and Multiple Correspondence Analysis (MCA) [Greenacre 2007]. The histograms are used to show a coarse-grain view of the results: the frequency of only one Value level (Required) over the subjects’ answers. A more comprehensive view of the results is presented via MCA. MCA is a standard and very convenient statistical procedure to visualize in a two-dimensional space the associations among the levels of three or more categorical variables; in our case, Activity, Category, and Value. Though rarely applied in software engineering, MCA is particularly useful for three or less variables and when variables show many levels as it can provide a global view of the data facilitating interpretation.

MCA produces a symmetrical variable plot where each point represents a given level of a categorical variable. To understand and interpret MCA results, the key principle is that the distance among points on that two-dimensional plot is a meaningful measure of the degree of association of the levels across variables. Therefore, when levels of different variables appear close in the plot it implies they are strongly associated. Moreover, when all the levels of the same variable appear close to each other, we can
conclude there is a small difference among them and that, as a result, the variable has a low impact on the other variables. We meet the conditions under which MCA is an appropriate and useful technique [Greenacre 2007] as we are only considering three discrete variables but with many levels (13 categories * 5 activities = 65 levels). Furthermore, we have a minimum of eight observations in each level combination of these variables. In order to analyze the interaction effect between Activity and Category, we applied interactive coding analysis [Greenacre 2007], which consists in creating a new variable having as levels all the combinations of levels for Value and Activity. See [Greenacre 2007] for further details on how to analyze interaction effects via MCA.

For statistically testing the impact of Activity and Category on Value we applied logistic regression for ordinal response variables (Value in our case) with Activity, Category, and their interaction term as explanatory variables [Hosmer and Lemeshow 2000]. The significance of the effect of each explanatory variable is tested using the standard Likelihood Ratio Chi-square test [Hosmer and Lemeshow 2000]. We meet the data requirements as the sample size is large enough, with 11,000 observations.

In order to analyze the difference between the results of the controlled experiment and its replica we apply two independent MCA analyses based on their two respective data sets. We then plot the results of both analyses in one MCA Symmetric Variable Plot to assess the variation in positions for all Category and Activity levels. The higher the distance for each pair of points corresponding to the same level but a different experiment, the more inconsistent the results. We want to assess whether the main conclusions are the same for both experiments.

5.1.2 Results

Table 3 describes the results of MCA in a standard form: the coordinates of each level of each variable in the symmetric variable plot, as depicted in Figure 4. In Figure 4, F1 and F2 are the coordinates in the plot produced by MCA [Greenacre 2007]. The “Required” Value level (VR) is in the top-left quadrant and the closer the Category and Activity levels to this point, the stronger their statistical association with it. For example, category Related Requirement (CatRR) is strongly associated with value Required (VR), therefore suggesting that “related requirements” is a very important piece of design rationale information. Similarly, category Related Artifact (CatRA) is closely associated with value Useless (VU), thus suggesting that “Related artifacts” is not a very relevant piece of information. Figure 4 also shows Activity levels to be much closer to the center, thus
showing that Activity does not explain nearly as much of the variation in Value as Category does.

Table 3: Abbreviations and results of Multiple Correspondence Analysis (MCA).

<table>
<thead>
<tr>
<th>Variable</th>
<th>Treatment</th>
<th>Abbreviation</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue</td>
<td>CatIs</td>
<td></td>
<td>-28.225</td>
<td>3.267</td>
</tr>
<tr>
<td>Decision</td>
<td>CatD</td>
<td></td>
<td>-34.549</td>
<td>46.687</td>
</tr>
<tr>
<td>Status</td>
<td>CatS</td>
<td></td>
<td>-4.362</td>
<td>-18.965</td>
</tr>
<tr>
<td>Assumptions</td>
<td>CatAs</td>
<td></td>
<td>-8.528</td>
<td>-33.270</td>
</tr>
<tr>
<td>Constraints</td>
<td>CatC</td>
<td></td>
<td>18.700</td>
<td>0.828</td>
</tr>
<tr>
<td>Positions</td>
<td>CatP</td>
<td></td>
<td>-15.078</td>
<td>-12.607</td>
</tr>
<tr>
<td>Argument</td>
<td>CatAr</td>
<td></td>
<td>-19.953</td>
<td>-3.181</td>
</tr>
<tr>
<td>Implications</td>
<td>CatIm</td>
<td></td>
<td>8.872</td>
<td>-3.979</td>
</tr>
<tr>
<td>Related decisions</td>
<td>CatRD</td>
<td></td>
<td>-9.133</td>
<td>-5.353</td>
</tr>
<tr>
<td>Related requirements</td>
<td>CatRR</td>
<td></td>
<td>-20.438</td>
<td>15.138</td>
</tr>
<tr>
<td>Related artifacts</td>
<td>CatRA</td>
<td></td>
<td>37.141</td>
<td>9.952</td>
</tr>
<tr>
<td>Related principles</td>
<td>CatRP</td>
<td></td>
<td>22.678</td>
<td>-30.388</td>
</tr>
<tr>
<td>Notes</td>
<td>CatN</td>
<td></td>
<td>52.876</td>
<td>31.902</td>
</tr>
<tr>
<td>Activity</td>
<td>Required</td>
<td>VR</td>
<td>-74.422</td>
<td>41.726</td>
</tr>
<tr>
<td></td>
<td>Optional</td>
<td>VO</td>
<td>-3.504</td>
<td>-77.388</td>
</tr>
<tr>
<td></td>
<td>Useless</td>
<td>VU</td>
<td>83.001</td>
<td>26.514</td>
</tr>
</tbody>
</table>

Figure 4: Graphical results of Multiple Correspondence Analysis.
Regarding the interaction effect between Activity and Category, because the new variable created for the interactive coding analysis has a very high number of levels (65), its plot on a single figure would have been too complex to interpret. Therefore, in order to facilitate the interpretation of results, we distributed the MCA interaction results in 13 distinct figures (Figure 5), each figure reporting on one Category level at a time and its combinations with the five Activity levels. So each figure reports eight points: the three levels for Value (see Table 3) plus the five levels for Activity when combined with a particular Category level. As guidance for interpretation, recall that the more spread these five points, the stronger the interaction effect between Activity and the figure’s specific Category level.

(a) DRD category Argument

(b) DRD category Assumptions

(c) DRD category Constraints

(d) DRD category Decision
Accepted for publication at ACM Transactions on Software Engineering and Methodology

(e) DRD category Implications

(f) DRD category Issue

(g) DRD category Notes

(h) DRD category Positions

(i) DRD category Related artifacts

(l) DRD category Related decisions
Figure 5: Interaction effect analysis through Multiple Correspondence Analysis.

Table 4 reports the results of the Likelihood ratio Chi-square test to assess the significance of the impact of Activity, Category, and their interaction on Value. Because p-values related to Category, Activity, and their interaction, are far below our level of significance $\alpha = 0.05$, both their main and interaction effects on Value are significant at a 95% confidence level. Figure 6 plots on a single figure the differences of MCA results between the experiment and its replica. Stars and circles depict the two, respectively. The higher the distance between a variable in the experiment and its replica, the higher the difference.

Table 4: Likelihood Ratio Chi-square test results of the effect of Category, Activity, and their interaction on the Value of a DRD information item.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Likelihood Ratio Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ChiSquare</td>
</tr>
<tr>
<td>Activity</td>
<td>45</td>
</tr>
<tr>
<td>Category</td>
<td>4251</td>
</tr>
<tr>
<td>Category*Activity</td>
<td>403</td>
</tr>
</tbody>
</table>
5.1.3 Discussion

Both Category and Activity have a statistically significant impact on Value. In Figure 4, the distance between levels of Category and Activity on one hand, and levels of Value on the other hand, vary significantly. The logistic regression analysis in Table 4 confirms statistical significance. According to Figure 4, all the levels of Activity are close to each other and near the center; this means that Category is a much stronger predictor than Activity on Value. This is confirmed by the statistical analysis in Table 4: the Chi-Square value of Activity is much lower than that of Category. There is a significant interaction effect between Activity and Category on Value. The logistic regression analysis in Table 4 shows that the interaction term between Activity and Category is statistically significant. However, according to Figure 5, when an interaction effect exists, Value changes only from Required or Useless to Optional (or vice versa) and never from Useless to Required (or vice versa). This means that, even when present, the interaction effect has a limited effect on Value.

Figure 6: Difference between Experiment and Replica through Multiple Correspondence Analysis.

Based on the analysis results, the categories of design rationale documentation can be divided into three groups:

- **High value and independent from the activity to support.** Categories that are very valuable, independently from the activity to support, include the description
of the decision (CatD), the Issue (CatI), and of the related requirements (CatRR). Note that in Figure 5-(d), Figure 5-(f), Figure 5-(n) their positions are near Required (VR) for all the activities.

- **Low value and independent from the activity to support.** Only one category shows little usefulness regardless of the activity to support: Notes. In Figure 5-(g) all activities are placed near VU (Useless). We recall that the “Notes” category contains any information that is not recorded in the other DRD categories. Because this category shows to be useless for all five activities, we can then conclude that the framework proposed by [Tyree and Akerman 2005] is comprehensive and no additional information needs to be added for documenting a design decision.

- **Ranging from high to low value according to the activity to support.** The remaining categories are only relevant for some of the activities to support. For instance, the category Positions (i.e., the description of the considered alternatives) provides value only when the activity to support is Decision verification or Wrong solution space. On the contrary, Positions is optional to support Impact evaluation; see in Figure 5-(h) that Activity 5 is near Optional (VO). This makes intuitive sense as, given new requirements, it is optional to know the set of considered alternatives; what is truly relevant is the chosen alternative.

There are only minor differences between the results of the controlled experiment and the exact replica. Based on Figure 6, the position of the 13 categories did not significantly vary. This is especially the case for those categories that were most strongly associated with Value levels and further away from the center. For example, CatD and CatRR remain close to VR. Visible variations are for those categories near the center that have no strong association with any Value level. This result suggests, for the main associations, that the results are consistent across the two experiments and therefore that the data can be merged to enhance statistical power.

5.2 R.Q. 2: How much effort can be saved by adopting a value-based DRD?

5.2.1 Analysis and Results

In the absence of evidence showing actual differences in required effort among information items, we assume that the number of items of a design rationale documentation to be a good indicator of the relative effort. To compute the expected effort savings in adopting a value-based DRD, we compute the percentage of information items considered relevant and valuable for a given activity (value-based DRD).
As explained above, a value-based DRD is composed of valuable information items only. An information item is defined as valuable according to a threshold on Value; in particular, a threshold on a given frequency of Required among subjects’ scores. Adopting a medium threshold (frequency of Required = 50%) results in documenting only the information categories that are more likely than not to support the readers (i.e., the information categories that have been perceived at least half of the times to be required by subjects). Though a medium threshold makes sense, we also analyze the expected effort savings in relation to other thresholds. Obviously, the lower the threshold, the higher the number of information items included in the value-based DRD, the higher the required effort, the lower the probability that a subject would require an information item that is not included in the DRD. We consider five thresholds on Value: Lowest = Value > 0%, Low = Value > 25%, Medium = Value > 50%, High = Value > 75%, Highest = Value = 100%.

Figure 7 describes the number of information categories in the value-based DRD according to both the threshold on Value and the activity to be supported (e.g., Impact evaluation). Figure 7 is composed of a histogram and a table. In the histogram, columns show, for each supported activity, the number of DRD information categories associated with each frequency threshold for Required. The table provides the exact number of information categories for each percentage threshold. For instance, a value-based DRD for the Decision verification activity (third column) with a medium threshold (frequency of Required = 50%, third row) requires only four information categories out of 13.

5.2.2 Discussion

Based on the third row in Figure 7, the number of information items included in a value-based DRD, tailored for a medium threshold, varies from 4 (third and fifth columns) to 8 (first column) depending on the activity to support. Therefore, the effort savings in documenting only valuable information items (i.e., categories that is more likely to support the reader in a given activity) varies from a maximum of 70% (Decision verification and Impact evaluation) to a minimum of 40% (Checking wrong solution space). On average across activities, a value-based DRD includes only 6 information items out of 13, that is 46% of a full documentation.

The significant reduction of information items to document is therefore expected to mitigate the effects of inhibitors that are currently preventing practitioners from documenting the rationale of their decisions. Though these results are generalizable only
to the five activities we have adopted, they have practical relevance since these activities are of widespread importance.

![Figure 7: Amount of information items in a tailored DRD according to different thresholds on Required.](image)

**5.3 Threats to Validity**

In the following subsections we provide insights regarding the validity of the aforementioned results based on the possible threats to validity as suggested in [Falessi, et al. 2010, Wohlin, et al. 2000].

**5.3.1 Conclusion Validity**

Conclusion validity concerns the reliability of the observed relations among the experimental variables [Juristo and Moreno 2006, Wohlin, et al. 2000].
Reliability of Measures: Regarding research question 1, we considered the use of a subjective measure of Value of a DRD information item, i.e., the subjective opinion of the experiment subjects. However, in the current industrial and research practice, the architecting activity is inherently a human endeavor and hence is subjective in nature. For instance, it is very hard to objectively judge if a design decision is correct or not [Falessi, et al. 2010]. In our specific context, in the absence of any available objective measure of the Value of DRD information category, it is estimated by averaging subjective values. Regarding research question 2, the magnitude of effort reduction is computed as the difference between the number of information items of a full DRD versus a customized DRD. But the actual effort saved by not documenting a DRD information category is significantly related to the technology in use where one category can be particularly easy or hard to document. Therefore, measuring the reduced effort as the number of reduced information items to document has the advantage to provide results applicable to a wide range of contexts.

Replica independence: There is a low risk that researchers biased the replica results since the replica was enacted before executing a complete data analysis of the original experiment. Therefore, the original experiment results were unknown and could not be communicated to the subjects of the replica. Second, the experiment results would have been in any case difficult to communicate given the very high number of levels for variables (13 DRD categories * 5 activities * 3 Value levels = 195). In other words, it would have been difficult to influence (consciously or unconsciously) the subjects of the replica to provide a specific value, that was observed in the experiment, for a specific DRD category, in a given activity.

5.3.2 Internal Validity

Internal validity is the degree to which conclusions can be drawn regarding the causal effect of the independent variables on the dependent variables [Juristo and Moreno 2006, Wohlin, et al. 2000]. The level of internal validity related to our experiment should be considered high since we balanced the order of activity executions and we randomized the application of treatments.

5.3.3 Construct Validity

Construct validity is the degree to which the independent variables and dependent variables accurately measure the concepts they purport to measure [Juristo and Moreno 2006, Wohlin, et al. 2000]. This was addressed in multiple ways.
Mono-operation bias: This was the threat with highest priority. To prevent experimental results from being too specific to the units of analysis (decisions), we adopted a set of twenty-five different decisions. Hence, since all the subjects performed all (five) activities, for each dependent variable we have a distribution composed of two hundred and fifty data points.

Restricted generalizability across constructs: Regarding the proposed value-based DRD, the present study analyzes its efficiency (i.e., documentation effort reduction) without considering effectiveness. In other words, it may be unclear whether a reduced documentation negatively affects the output quality of the activity it supports. In general, the quality of a design decision (i.e., the activity output) is difficult to measure [Falessi, et al. 2010]. In this study, we indirectly and implicitly estimate effectiveness as the probability that a subject requires an information item that is included in the customized documentation.

Hypotheses guessing and experimenter expectancies: During the training sessions, we did not convey our expectations and hypotheses to participants.

Evaluation apprehension: We informed the subjects that they would not be personally evaluated based on their answers.

Low motivation: Subjects were not pressured to participate in the experiment. We clearly explained that their course grade would not be related to their presence or performance during the experiment. The subjects would not provide random answers because they knew that the examiners would ask explanations during the exam of the Empirical Software Engineering course. This is an approach we have successfully adopted over several years in other experiments [Cantone, et al. 2003, Falessi, et al. 2010, Falessi and Cantone 2006]. We observed a high level of commitment and concentration, as visible on the video we posted at: http://eseg.uniroma2.it/DDRD-Experiment-December2006.zip.

5.3.4 External Validity

External validity is the degree to which research results can be generalized to other settings than the experiment one. Since the use of students as subjects may be interpreted as an important threat to external validity, we carefully described the experimental process by following standard guidelines [Jedlitschka, et al. 2008] in order to enable replications in different and possibly more realistic contexts. The following list discusses how we addressed various external validity threats, with an emphasis on the experimental subjects and objects.
**Experimental Subjects:** The main reason why we chose master students as the subjects of our experiment is because we could properly train them to use the design rationale documentation, documentation goal much harder to achieve with busy practitioners over a short period of time. Though the use of students may appear as serious threat to validity, a number of studies actually show that differences between students and practitioners may not be relevant. This is the case, for example, of studies in the context of requirements selection [Svahnberg, et al. 2008], assessment of lead-time impact [Host, et al. 2000], and mental representation of programs [Holt, et al. 1987]. Whether this is the case for DRD activities is an open question. Furthermore, we must note that about half of the subjects in our experiment had significant professional experience. In addition, most computer science and engineering courses include practical exercises or projects and therefore master students may not be so different from junior practitioners as observed in [Sjøberg, et al. 2001, Sjøberg, et al. 2001]. Last , “the variations among students and variations among professionals may be so large that whether the person is a student or a professional, may just be one of many characteristics of a software engineer”[Sjøberg, et al. 2002].

Moreover, our experiment was designed to: i) ensure that subjects had a reasonable ability and level of confidence in the task to perform (see Section 4.7), and ii) perfectly balance the assignment of treatments to subjects; in this way, every treatment has been applied by every subject and thus at all levels of experience.

We neither measure nor control the variable “experience” of subjects. The main motivation for this decision is that experience has a broad and uncertain meaning in our context. Therefore, any experience measure would probably be inadequate and would therefore provide misleading results. Moreover, the experiment is already complex as it is given the high number of input variables and levels: 13 DRD categories x 5 Activities x 3 Utility levels= 195 levels. Therefore adding a variable would make the MCA results less understandable. Finally, since an information item represents a piece of knowledge, it is obvious that people with less experience or knowledge tend to require more information items than more experienced people. Therefore, more experienced readers require less documentation, which results in greater effort savings for documenting only the required information items. Consequently, assuming students have less experience than practitioners, the effort savings in adopting a value-based DRD should logically be expected to be higher in industrial contexts than in our experimental setting. In other words, effort reduction brought by value-based DRD is expected to be conservative when using students as experimental subjects.
Experimental material: In general, it is not possible to eliminate all the threats to validity; experimenters should prioritize them according to their context and research question to address [Wohlin, et al. 2000]. In particular, there is always a tradeoff between realism and control [Dzidek, et al. 2008]. We deliberately chose to use an artificial artifacts given the time constraints of a controlled environment and the need to focus only on the effects of DRD on decision-making. Regarding the representativeness of the units of analysis, the decisions involved in the tasks were complex since they were characterized by several, opposite and interrelated objectives, as in realistic contexts. We decided to involve a large number of decisions (i.e., twenty-five, five per role) in order to study the impact of activities in a somewhat independent manner from the specifics of the decisions to be made. In other words, we defined a large number of varied decisions to achieve better external validity.

Experimental task: This work involves individual rather than team decision-making and does not account for decision making based on past releases or iterations of the system. But we would expect an even greater need for explicit decision rationale under such conditions. In other words, we expect the effects observed in our experiment to be minimal or conservative and this is why we recommend, in the next section, possible subsequent steps to perform industrial case studies where team decision-making based on information from past releases would take place.

6. THE WAY AHEAD

The maturity of empiricism in software engineering can be considered low when compared to other scientific disciplines and the software architecture sub-field is not an exception. During the last two decades, empirical software engineering research has gained in maturity and experience. For instance, the application of empiricism provided significant advances in the area of software economics [Boehm 1981, Boehm and Sullivan 2000], software quality [Boehm, et al. 2010, Shull, et al. 2006] and value-based software engineering [Biffl, et al. 2005]. During the same period, software architecture has emerged as an important field of software engineering for managing the development and maintenance of large, software-intensive systems. Historically, most advances in software architecture have been driven by talented people and industrial experience, but there is now a growing need to systematically gather empirical evidence rather than just rely on promotional anecdotes or rhetoric [Oates 2003]. Performing valid empirical software architecture research is still very hard due to the existence of several challenges
caused by the inherent complexity of the software architecture discipline [Falessi, et al. 2010].

Using a complex empirical setting, we report in this paper two controlled experiments investigating if the value of information items in DRD documentation is influenced by its category and the activity it supports. Despite encouraging results, how to make DRD cost-effective in practice is certainly a complex question that cannot be answered by any single (empirical) study, such as the one reported here.

In a controlled artificial setting, we cannot perform any cost-benefit analysis because we would need to account for several practical issues typically arising in real projects (as detailed below). But we can assess whether selective DRD can be potentially beneficial and is worth studying further in more realistic contexts. To be realistic, for any study, time and budget constraints make it impossible to run a fully realistic investigation accounting for all variables that can potentially influence DRD cost-effectiveness. The research community should follow an iterative process, of creating and analyzing evidence, with the two-fold aim of confirming well-defined theories and suggesting new ones [Basili, et al. 1999, Ragin 1989, Wohlin, et al. 2000]. Our controlled experiments, like most in software engineering, took place in an artificial environment to prioritize control over realism. There is significant work that remains to be done before the DRD can achieve widespread adoption. This work nevertheless paves the way for future coordinated and integrated research efforts. Our controlled experiments can be seen as a first, fundamental step; but a collective and coordinated effort must be applied by the research community [Basili, et al. 1999]. Large effort should be spent in integrating results and planning subsequent observations. In general, future controlled experiments should continue our work aiming at reducing the scope of the variables influencing DRD cost-effectiveness. Large scale observational study should follow and, benefiting from reported experiments, should focus on the variables shown to be significant, thus keeping their time and budget under control. For example, future studies may be restricted to only the DRD categories that have shown to be valuable in our experiments for certain activities. More specifically, future studies can proceed, based on the presented results, by following three orthogonal directions: independent variables, dependent variables and realistic context.

Independent variables: enlarging the scope. Because our study was the first one exploring the variables that can possibly impact the value of DRD, we focused on the most intuitive aspects: supported activity and DRD category. Future studies can
complement the present study by analyzing whether the following independent variables impact DRD cost-effectiveness:

**Phase of the software development lifecycle:** The time in which a decision take place can impact significantly its value. Decisions taken in the early phases of the design have intuitively a high impact on the overall project because subsequent decisions must be based on it. For instance, once a middleware has been selected, the choice of the programming language is partially restricted.

**Decision-making technique:** Decision-making in software design is a reasoning process that consists in analyzing requirements and business/contextual constraints, choosing an alternative among feasible options, and assessing the solution as a whole. This process can be performed by individual subjects or teams [Falessi, et al. 2006], by following or not a structured technique [Falessi, et al. 2011]. This paper focused on individual decision-making and without the use of any structured technique. Future studies may investigate which DRD categories are required by which team-based decision-making techniques. Moreover, the reasoning process characterizing decision-making in software design should be thoroughly investigated [Tang 2011].

**Supported activities:** This study focused on five specific activities in which a designer may benefit from DRD (Section 4.4). There may be further activities that have not been investigated in this paper and are relevant in specific contexts. Examples of such activities are reported in [Kruchten, et al. 2006].

**Complementary documentation:** The importance of a piece of information in the DRD may vary according to the information already present in complementary documentations such as traceability documents and architectural views. Documentation should be viewed as an investment and its cost-effectiveness should be investigated by analyzing also the interactions among the different types of documentation. Overlaps should be minimized and optimizing the set of used documentation should be investigated.

**Application context:** Software is used in a wide spectrum of contexts, from entertainment (e.g., games), to safety-critical systems (e.g., air traffic control systems), to applets for smartphones. Some of the activities supported by DRD may be more or less vital according to the application context. Since the value of DRD information items has shown to be dependent on the activity it supports, the same DRD information item could be more or less valuable based on the application context.

**Realistic dependent variables: the business value of DRD.** Though our experiments showed that a value-based DRD resulted into significant cost reduction, we were not in a position to study the business value of enforcing the production and use of such
documentation. This would require to measure both the real cost and the benefits of DRD in realistic contexts. This is elaborated below.

Measuring cost: Our study showed that the size of DRD can be significantly reduced without unduly compromising its benefits. However, the high-value DRD categories could also be the most expensive ones, thus making effort savings less significant. However, given what we know now, that is very little, the first question to ask is whether customized DRD is even worth studying. Moreover, we note that the cost of a DRD category strictly depends on the technology supporting it (e.g., software tools). For example, a DRD category can be inexpensive when using an advanced technology that is able to automatically record decision rationale, whereas the same DRD category can be very expensive in other contexts when manual tasks are involved. Therefore, in this paper we focus on the DRD value and leave it to the reader to assess, in context, her own cost-benefits tradeoffs.

Measuring economic benefits: Our main purpose was to reveal the magnitude of potential differences in the values of an information item according to its category and the activity that it supports. Therefore, in this paper we measured the value of DRD categories via the theory of subjective value, see Section 4.4. We see this as a first important step to pursue research on value-based DRD. Further analyses must be performed in real settings to understand when a good design decision is actually important to deserve the investment of documenting its rationale.

Measuring locality of DRD costs-benefits: The present study makes simplifying assumptions regarding where the DRD cost and benefits take place, mainly because there is no reported empirical data on that topic. Specifically, here we assume the DRD producer as incurring cost for documenting rationale information and the DRD consumer as incurring value. If this assumption holds, DRD would have a dismal future regardless of the savings provided by a value-based customization strategy. Future field studies should investigate the actual “locality” of both cost and benefits for DRD. This issue can’t be addressed in controlled settings but only via an industrial case study.

Context: from small and artificial to large and real. The high cost of conducting empirical studies tends to create a tension between control and realism; researchers always struggle to find the right balance. Because analyzing all the possible interesting variables in real development context is too expensive to be feasible, it is usual to first proceed with controlled studies in artificial settings. Future studies can explore the use of design rationale information in industrial settings, based on what was learned in our controlled experiments. Specifically future studies may improve on the realism of the
following aspects while focusing on the DRD categories that showed, for specific activities, to be valuable:

Real objects: Our study intentionally made use of an artificial object of study, but with as high a level of complexity as possible given the practical constraints of our controlled experiments. Decision-making and the value provided by DRD, should be observed and analyzed when applied in the context of realistic artefacts.

Real subjects: In this study, the use of students, though fully trained and competent, was opportunistic rather than intended. Further studies should involve professional designers, though existing studies suggest that adequately trained students perform as well as junior or intermediate consultants (see Section 5.3.4). Professional subjects are an expensive resource, whose cost is a significant barrier to carrying out controlled empirical studies.

7. CONCLUSION

Documenting software design rationale is a strongly recommended practice: recording a designer’s line of reasoning makes it possible to revisit it later in order to assess it, to approve it, or more simply to learn from it, with regards to either the system being designed or the decision process itself. But the methods and tools proposed so far to document design rationale have not been very successful; they all tend to try to maximize the benefits for the consumer of the rationale, at the expense of the producer—that is the designer—and they are therefore too onerous for systematic industrial use. In order to facilitate the industrial adoption of rationale documentation we investigated how to reduce the amount of effort required from the producer side.

The underlying intuition of our work is that design rationale documentation can potentially be used in many activities (e.g., what if analysis, avoiding design erosion) but not all of them are enacted in every context. Therefore, documenting all the information items that support all the activities becomes somehow ineffective from a cost-benefit point of view; this eventually inhibits practitioners to document the rationale of their design decisions. The key idea is that design rationale documentation should be introduced (for adoption) only to support those activities which are particularly hard to perform with the usual procedures and documentation.

Such value-based design rationale documentation is supposed to contain only the information items that will be required, with a significant probability, to support a given activity. Therefore, the aim of the paper was two-fold: 1) validating the feasibility of the proposed customization strategy, 2) assessing its effectiveness by estimating the expected
effort savings. Specifically, in this paper, we report on two controlled experiments that investigate whether the decision to document or exclude an information item in DRD should depend on its category and the activity it aims to support. An efficient customization is essential because it provides a realistic means for mitigating the extra effort inhibitor of documenting design rationale. Our experimental results show that:

- The value of an information item is significantly affected by its category; moreover, different categories have different values for different activities. Therefore, it is possible to prioritize the information items to document according to the category and the activities that design rationale documentation targets.
- On average, across activities, a value-based design rationale documentation contains 46% of the information items in the full documentation. The maximum effort reduction provided by a value-based design rationale documentation is 70% and concerns the “decision verification” activity: only 4 out of 13 information categories have been shown to be required more than 50% of the time.
- Our experiment replication, performed in a different setting and geographical location, provided similar results to the original controlled experiment; this suggests that all the important variables have been taken into account in our experiment design.

Our results suggest that a value-based design rationale documentation facilitates its practical adoption by focusing and investing documentation effort where it is most needed. This approach can be easily applied regardless of the documentation type and medium; it consists in documenting only the information items that are expected to be valuable for a target activity.

Despite the observed significant effort savings provided by a value-based customization of DRD in our experiments, we cannot assess its cost-benefit as this would require accounting for several practical issues that can only be studied in real project settings. At this stage, based on the presented results in this paper, we can however observe that selective, value-based DRD is promising and worth studying further in more realistic contexts.

8. REFERENCES


9. APPENDIX A: FRAGMENT OF EXPERIMENT MATERIAL

The present section provides a fragment of the material adopted during the experiment. All the material we used was in native language, that is Italian and Spanish in the original experiment and replica, respectively. This section reports the experimental material in English to promote the understandability. The description of the project and its high level requirements, aka. the vision document in RUP, is pretty large in size. Therefore, Section A1 reports only a summary of this document. Section A2 reports an example of DRD, Section A3 reports an example of change, and Section A4 reports an example of activity to perform.

9.1 A.1. Summary description of the project

The system under consideration consists of a software system for managing a service system for the transport of passengers by bus. The system offers features: the Manager of the transport service, the driver and passenger of a vehicle. The system is designed to operate in three main parts:

1. Central system, which manages the maintenance of vehicles and shall, at the time of execution, input to the rest of the sites of the system will converge and can be analyzed all the system information from other sites and is characterized by an ability to calculation that may supposed unlimited.
2. Totem, in various instances, placed along the route of the buses, at bus stops for passengers and helps them make reservations for the routes you want, pay for tickets, check times of departure and arrival.

3. Intelligent bus, consists of an onboard computer, a temperature sensor and humidity, a video camera and a system of marking. The onboard computer manages temperature and humidity inside the bus and, according to passenger bookings, giving directions to the driver. It is noted that the bus is intelligent, and the totem only communicate directly with the central system and not between them.

The system must handle emergency situations such as collisions, fires, accidents and sudden illnesses of passengers or the driver. In routine conditions of operation and service, the intelligent bus communicates with the central system, at each stop, using the protected access point (Bluetooth or wi-fi), offered by each totem in emergency situations, the ' Intelligent bus will communicate via cellular network. A bus provides some services to its passengers, who can access it via their mobile phone or handheld PDA (Bluetooth or wi-fi). The computing capacity on-board computer is similar to that of a laptop.

The initial design is made not only by the system, only the central totem of about 150 and about 40 buses. The whole performance is, for quality of services and so on, paid to the needs of an average citizen, it expects to recoup the costs in 5 years.

9.2 A2. A Design Decision and its Rationale Documentation

- Issue: Determine the operating system to install on each on-board computer.
- Decision: Windows.
- Status: Pendent
- Assumptions: Such a system should be installed on all buses. Not all features of a standard operating system are necessary, on the contrary, some of the features offered could be wasting memory and / or computing power.
- Constraints: None.
- Positions: Linux, Windows XP, Sun Solaris, Symbian, Mac OS X.
- Argument: Medium level of reliability.
- Implication: All software must be Windows compliant.
- Related decisions: C# has been decided and approved as the standard programming language for the on-board computers.
- Related requirements: Reliability and support should have a medium level.
- Related artifacts: None.
Related principles: The development organization prefers the use of standard and well known technologies over state of the art promising technologies.

Notes: None.

9.3 A.3 New Requirement or Change
The availability of the memory has increased by 80%. The required level of reliability has become a low. The requested level of support is now high.

9.4 A.4 Activity to perform
Detecting conflicts between new requirements and an old decision. Is the old decision still valid for the new set of requirements?