Architectural Decision-making as a Financial Investment: An Industrial Case Study

Areti Ampatzoglou[1], Elvira-Maria Arvanitou[2], Apostolos Ampatzoglou[2], Paris Avgeriou[1], Angeliki-Agathi Tsintzira[2], and Alexander Chatzigeorgiou[2]

[1] Department of Computer Science, Institute for Mathematics, Computer Science and AI, University of Groningen, Netherlands

[2] Department of Applied Informatics, University of Macedonia, Greece

areti.ampatzoglou@rug.nl; e.arvanitou@uom.edu.gr; a.ampatzoglou@uom.edu.gr; paris@cs.rug.nl; angeliki.agathi.tsintzira@gmail.com; achat@uom.edu.gr

Context: Making architectural decisions is a crucial task but also very difficult, considering the scope of the decisions and their impact on quality attributes. To make matters worse, architectural decisions need to combine both technical and business factors, which are very dissimilar by nature.

Objectives: We provide a cost-benefit approach and supporting tooling that treats architectural decisions as financial investments by: (a) combining both technical and business factors; and (b) transforming the involved factors into currency, allowing their uniform aggregation. Apart from illustrating the method, we validate both the proposed approach and the tool, in terms of fitness for purpose, usability, and potential limitations.

Method: To validate the approach, we have performed a case study in a software development company, in the domain of low-energy embedded systems. We employed triangulation in the data collection phase of the case study, by performing interviews, focus groups, an observational session, and questionnaires.

Results: The results of the study suggested that the proposed approach: (a) provides a structured process for systematizing decision-making; (b) enables the involvement of multiple stakeholders, distributing the decision-making responsibility to more knowledgeable people; (c) uses monetized representations that are important for assessing decisions in a unified manner; and (d) enables decision reuse and documentation.

Conclusions: The results of the study suggest that architectural decision-making can benefit from treating this activity as a financial investment. The various benefits that have been identified from mixing financial and technological aspects are well-accepted from industrial stakeholders.

1. INTRODUCTION

One of the most important tasks of software architects is to make ***architectural decisions*** [16]. This is a particularly challenging task since architecture decisions often affect large parts of the system and impact quality attributes; consequently, one needs to get them right from the beginning as they are very hard to change in the future [22]. The difficulty of these decisions is further aggravated by having to consider not only technical factors (such as reusability, maintainability, testability, etc.), but also market and business factors (such as time-to-market, achieved market share, etc.). However, it is often hard, if not impossible to weigh these different factors against each other, e.g., comparing the impact of a decision on coupling and cohesion with the impact on time-to-market. Being forced to compare "apples and oranges", architects often make ad-hoc, intuitive judgement calls, which are prone to bias [35].

In this paper, we aim at addressing the issue of taking heterogeneous factors into account and comparing them against each other when making architectural decisions. Specifically, we propose an approach, named ADMIT, that looks at this problem from a ***financial*** perspective, by treating decisions as ***investments*** through a ***cost-benefit analysis***. This is achieved by expressing all inputs in monetary terms, and then comparing the financial benefits obtained by applying the architectural decision, against its costs. The main benefits from treating architectural decisions as financial investments are: (a) the ability to mix and aggregate technical and business factors of the decisions, e.g.,

improvement in market share, cost to write code, and effort required in future maintenance activities; and (b) the power to express all factors in a uniform way, especially using monetary terms, which are more easily perceived by higher management than purely technical ones.

The proposed approach relies on defining a set of costs and benefits related to the architectural decision under study, and comparing them to assess if the decision should be taken or not. To aid software architects with the selection of costs and benefits, the proposed approach comes with a set of default cost and benefit models (see Figure 1). Specifically, the ***modelling of costs*** relies on the well-established PAF (Prevention-Appraisal-Failure) cost model—see Section 3.1 and Section 3.2; whereas, the ***modelling of benefits*** relies on the architectural quality business goals, as defined by Kazman et al. [12] (see Section 3.3). In addition to these default models, a software architect is of course free to introduce custom models, depending on the relevant technical and business factors at play. We note that some costs or benefits (especially quality-related ones) might be difficult to monetize; for such cases, we expect the decision-maker to provide (at best) an informed estimate. Subsequently, the models are instantiated, i.e., estimated values are assigned to each factor. As already mentioned, all cost and benefit models provide their outcome as a currency value, allowing them to be aggregated into a total cost and benefit. To ease the applicability of the ADMIT approach, we have developed an accompanying tool. The tool provides a level of automation but the approach is not fully automated: in practice, a number of costs and benefits are evaluated by experts.

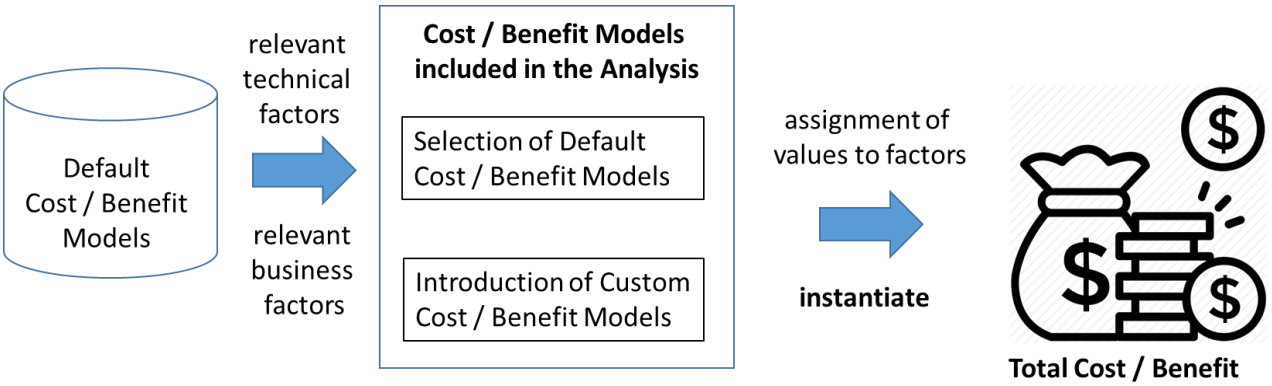

Figure 1: Overview of the Proposed Approach

We note that ADMIT is not a full-fledged and standalone architectural decision-making approach, in the sense that it only considers individual decision alternatives, while it leaves out the typical elements of the decision (e.g., design issue, alternatives, relations with other decisions, decision status, etc.) – see for example the work of van Heesch et al. [33] or Kruchten et al. [17]. Instead, ADMIT *supports cost-benefit assessment of individual decisions* and can be used in combination with other architectural decision-making approaches and tools (it is not meant as a replacement for them). For example, each of the alternative decisions can be assessed with respect to cost-benefit with ADMIT, and then the optimal one can be further assessed through another decision-making process that considers the interplay between decisions. If we consider the generic architecture design process by Hofmeister et al. [37], ADMIT can be positioned within the activity of *Architectural Evaluation*, as it allows a cost-benefit evaluation of "*candidate architectural solutions*".

Furthermore, we note that the proposed approach cannot (realistically) be applied to all decisions, due to the effort required to build cost/benefit models. Thus, it is expected to be useful only for the most important decisions, especially high-risk or high-impact decisions. However, the identification of high-risk or high-impact decisions does not fall within the scope of ADMIT, since it is part of risk management; there are specialized approaches, such as

ATAM [2] or RCDA [26], which aim at identifying risky architectural decisions. Finally, the financial perspective of ADMIT does not dictate narrowing its scope to decisions in the financial context of a company, in the sense that a wide-range of architectural decisions have a strong financial impact on the software and the organization.

To validate the applicability of the proposed approach and also evaluate its accuracy and usability, the approach has been applied in a real-world industrial case from the low-energy embedded systems domain, and has been empirically evaluated by practitioners. In particular, a case study was designed and executed, in which four practitioners discussed their experiences in architectural decision-making, identified benefits and limitations on their current way of working, applied the proposed approach in the context of their day-to-day job, and evaluated its perceived usefulness and usability. To make the study as unbiased as possible, we have employed method triangulation, using four data collection methods (namely: interviews, focus groups, questionnaires, and task analysis). The results have been analyzed using well-accepted methods for qualitative studies [30].

The rest of the paper is organized as follows: in Section 2, we present related work on architectural decision-making. In Section 3, we provide background information that is necessary for the better understanding of the paper. In Section 4, we present the proposed approach and the accompanying tool, which we validate in Sections 5 (study design) and 6 (results). The obtained results are discussed in Section 7: (a) with respect to existing literature; (b) future work opportunities; and (c) implications for researchers and practitioners.

2. RELATED WORK

In this section, we present related work that has been published in the field of architectural decision-making, both in terms of methods that employ cost-benefit analysis, and in terms of tool support. Note that, since we were not able to identify tools that support decision-making based specifically on cost-benefit analysis, we provide a more generic discussion on tool support for architectural decision-making. We note that the main advancements compared to existing studies are discussed at the end of this section, and not per paper.

Cost-Benefit and Risk Analysis for Architectural Decision-Making. Kazman et al. [12] propose the use of Cost-Benefit Analysis Method (CBAM), which is applied for assessing candidate solutions to an architectural problem. In particular, the authors suggest that based on the business goals of the software, several architectural decisions can be evaluated. As a first step for applying the process, an estimate of the cost of applying each solution is made by the software architect. Second, the architect selects the quality attributes of interest and specifies the anticipated quality improvement, through a concept termed utility. Finally, these quality gains are transformed to benefits, which can be compared to the cost of each alternative solution. The main difference between CBAM and our approach, is that the former uses scenarios to calculate benefits while it does not elaborate on the calculation of costs; our approach instead focuses on cost estimation using PAF and provides default cost and benefit models. Also, the use of a unified currency unit is arguably more effective when interacting with business stakeholders. In the same line of research Lee et al. [18], adopt the Analytic Hierarchy Process (AHP) and the Analytic Network Process (ANP) methods to perform an elaborate cost-benefit analysis of architectural decisions. The proposed process consists of two phases: the development of a Scenario-based Architectural Strategy, and the application of the Cost Benefit Analysis to assess the existing strategies (i.e., design alternatives). Both processes (AHP and ANP) receive input from various stakeholders, so as to initially calculate the best, worst, current and desired values for the decision parameters. Next, based on the aforementioned input, each strategy is assigned with a weight, through pair-wise comparisons. Upon the calculation of costs and benefits, the Return of Investment (ROI) for each strategy is calculated and used for prioritization. Finally, Poort and Van Vliet [26] suggest that architecting can be treated as a risk and cost management discipline. The risk and cost are mapped to core architectural practices and principles, which in turn can be evaluat-

ed, based on their cost and risk values. Nevertheless, we need to note that the two aspects are not synthesized, and that most of the emphasis is given to risk, whereas our study focuses on decision costs and benefits. Such a point of view can help decision-makers to identify the key concerns that drive decisions, through a simple and objective way. The use of risk and costs (similarly to our approach) enables the communication with business stakeholders, who are used to these concepts, when performing decision-making. The proposed approach, namely Risk-and Cost Driven Architecture (RCDA) has been empirically validated through a survey on more than 150 architects. The results of the validation suggest that RCDA has a significant positive impact on their architecting work.

Tool Support for Architectural Decisions. Kruchten et al. [15] argue in favor of an explicit representation of design decisions and other factors that drive those decisions, as a means to facilitate a system's evolution. They presented an initial set of use cases on the documentation of Architectural Knowledge (AK), a general term that mostly focuses on design decisions and their rationale. They also stressed the importance of an efficient visualization of the ontology of design decisions.

In the work of Babar and Gorton [1], the authors develop PAKME, a web-based tool for Managing Software Architecture Knowledge, supporting services of storing, retrieving, and updating artifacts of architectural knowledge, as well as the generation of reports that represent the relationships between different architectural artefacts, explicate them or depict their positive or negative effects on each other. Capilla et al. [4] present their own web-based application, namely the ADDSS—Architecture Design Decision Support System, which captures and documents architectural design decisions. The tool also explicitly links these decisions, so it is possible to estimate the effect of adding, modifying, or removing a decision. SAW (Software Architecture Warehouse), proposed by Nowak and Pautasso [23], is an architectural decision management tool that aims at supporting co-located and distributed design teams, under the perspective of collaborative decision-making. SAW allows the architect to follow relevant architectural issues and appropriate solutions for those issues. Tang et al. [31] suggest AREL (Architecture Rationale and Elements Linkage), an architecture model that captures the relationship between Architecture Rationale (AR) and Architecture Elements (AE). The authors implement this model as an add-in for Enterprise Architect which supports architects in the design traceability and reasoning process. The Knowledge Architect tool suite, proposed by Liang et al. [21], aims at supporting collaborative architecting. To this end, the tool is based on a Knowledge Repository, as a central location for storing and retrieving. The tool suite has been validated in two industrial case studies and is considered to simplify Architectural Knowledge capturing and sharing.

All the aforementioned tools are analyzed in two subsequent studies alongside the state of practice in tool support for Architectural Knowledge (AK) management [32] [5]. The first of these studies provides a framework for the comparison of these tools, according to criteria based on literature and architecture activities [32]. The second study is an extension of the first several years later, proposing new taxonomies (based on tool features) and aiming at exploring new trends and developments by conducting an industrial case study [5]. However, to date, these tools have remained research prototypes and have not been adopted in industry.

More recently, Manteuffel et al. [22], have developed a decision documentation tool, namely Decision Architect, which is meant to fulfil the expectations of industry and that software architects would find easy to adopt. The validation of the tool has been performed through two industrial case studies. The results of this work show that architects are generally positive on adding a documentation tool in their daily routine and perceived the tool as useful and relevant. Additionally, Li et al. [20] in their book chapter on software economics suggest that the costs and benefits of architectural decisions (aiming at reducing technical debt) should be specified as an integral part of the decision documentation process. However, in that study the authors do not describe any process on how these costs and benefits should be retrieved, or synthesized.

Based on the above, our work extends the body of knowledge on architectural decision-making, by: (a) providing a set of default cost and benefit models to guide architects in more systematically selecting cost and benefits; (b) aggregating the results of more than one costs and benefits directly in monetary values, addressing the issues raised from the use of different units when assessing different quality attributes; (c) providing tool-support that guides the architect throughout the cost-benefit analysis.

3. BACKGROUND INFORMATION

In this section we present background information for understanding the underlying concepts of the proposed approach (costs and benefits). In particular, in Section 3.1, we present the Prevention, Appraisal, and Failure model that is used for modeling the *costs* of the approach. In Section 3.2, we provide a review on the uses of PAF cost models in software; these are used for building the default cost models of the proposed approach. Finally, in Section 3.3 we present a list of business goals in the domain of software engineering, which act as a basis and inspiration for the default *benefit* models presented in Section 4.

3.1 Prevention, Appraisal and Failure Cost Model

Quality cost models have been developed in the 1950s [13] as a means to serve the purpose of Total Quality Management (TQM) [27]. Feigenbaum used the term Total Quality Control (TQC) [8], and defined it as "*an effective system for integrating the quality-development, quality maintenance, and quality-improvement efforts of the various groups in an organization so as to enable production and service at the most economical levels which allow for full customer satisfaction*". The term "Quality Costs" lacks a broadly accepted definition, as it may be conceived either as referring to the costs of improving quality (quality costs) or the costs incurred due to low quality (poor quality costs) [13]. To address this issue, Juran proposed the economic conformance level (ECL) model in his 1951 Quality Control Handbook [10], which suggested the terms of conformance and non-conformance costs. Conformance costs refer to actions taken to ensure the production of high-quality goods, while non-conformance costs apply when resources are spent because the product does not conform to specifications. The Prevention – Appraisal – Failure (PAF) model, was introduced by Feigenbaum [9], and classified conformance costs into prevention and appraisal costs, while non-conformance costs were divided into internal and external failure costs [13].

- **Prevention costs** include costs that prevent or reduce the risk of defective products – e.g., process control, product and service design and redesign, supplier relations, audit and screening, preventive maintenance.
- **Appraisal costs** are incurred to evaluate the product's conformance to specifications – e.g., raw material inspection, in-process inspection, quality audit.
- **Internal Failure costs** are incurred by defective products, before the product reaches the consumer – e.g., scrap, rework, re-inspection of products.
- **External Failure costs** are those incurred by defective products, discovered after the product is delivered to the customer – e.g., warranty charges, complaint handling, and lost sales.

In the PAF model, while prevention and appraisal costs are increasing, failure costs tend to decrease, as more faults are revealed at an early stage. In any case, it is more costly to modify a defective product as it proceeds in the production phases. Total quality cost is defined by adding prevention, appraisal and failure costs, while the optimal quality level is the one minimizing total quality costs, i.e., the investment to quality is exactly the one required to achieve the least failure costs.

Figure 2 depicts the relationship between prevention-appraisal and failure costs, as well as the optimal (in terms of cost) level of quality. Although the PAF model has many drawbacks and limitations [13, 27], it is the most widely

used [13, 27] and "*almost universally*" accepted [25] quality cost model. From Figure 2 it becomes clear there is a specific point (in the conjunction of the two lines—Failure Costs and Cost of Prevention plus Appraisal) in which the Total Cost of Quality (CoQ) get minimized.

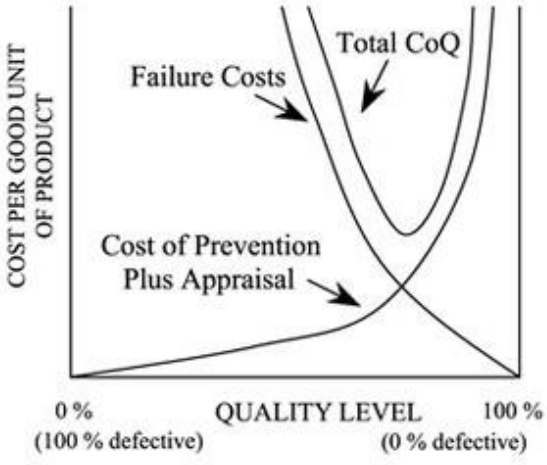

Figure 2: The Prevention, Appraisal and Failure Cost Model [24]

3.2 Prevention, Appraisal and Failure in Software Cost Assessment

In order to derive the default models for our approach, we conducted a meta-analysis on how the Prevention-Appraisal-Failure Cost Model has been applied in software engineering. The main goal of the meta-analysis was the identification of the most prominent types of costs related to the software quality assurance process. The meta-analysis used the primary studies of a systematic literature review [11]. More specifically, the first, second, and third author have gone through the primary studies to identify the most prominent costs, based on their frequency. In total, we looked at 87 studies from Karg et al. [11] that are related to prevention, appraisal, or failure costs. Within these studies, we identified 79 prevention, 114 appraisal, and 136 failure costs; these however were not unique, i.e., the same costs appeared more than once. Next, we went through the costs, so as to consolidate them and merge similar ones into a single title; this merging was driven by the fact that we would not be able to empirically evaluate the approximately 330 individual costs that had been identified. To achieve this, we followed the Open Card Sorting methodology, proposed by Spencer [30] as follows: (a) we identified themes (i.e., super-categories) from the costs as identified in the primary studies (without any processing); (b) we reviewed the themes to find candidates for merging; and (c) we defined the names of the final themes.

The final list of the consolidated costs in each category are presented in Table 1, accompanied with the frequency of primary studies, in which the cost has been referenced. For example, the "*Necessary Rework (Debugging, Quality Improvement)*", has been mentioned in 55 studies as a "*Failure Cost*" (F). We note that Table 1 maps the Costs to the respective categories as reported by Karg et al. [11]. Therefore, there are costs presented in multiple categories: For example, "*Costs for Hiring Quality Personnel*" is reported in all three categories by the primary studies but later on in the paper we classify it under "*Prevention Costs*", since it is the prevalent category, and is closer to our own interpretation of the specific cost. We have done the same for all costs that map to more than one category. As a means of confirming the list of default cost/models presented in Table 1, we have searched for similar studies (outside the field of software development), and contrasted the identified costs. We found only one similar study, conducted by Elizondo-Noriega et al. [7]; the top costs in each category in Table 1, are also found in the study of Elizondo-Noriega et al.

Table 1. Costs Identified in the Literature

Type	Specific Cost	#	Type	Specific Cost	#
Prevention	Costs for Hiring Quality Personnel	9	Approval	Cost for Performing Code Inspection	12
	Costs for Performing Meetings	9		Cost for Performing Design Reviews	9
	Costs of Training	8		Cost for Performing Unit Testing	9
	Increase Preventive Maintenance Effort	6		Cost for Using Quality Assurance Tools	6
	Cost of Producing Documentation	3		Cost for Performing Quality Audits	6
	Increase Cost for Project Planning	3		Cost for Performing Acceptance / Integration Testing	6
	Cost of Prototyping	3		Cost for Performing Metrics-Based Quality Assurance	5
	Cost for Reusing Proprietary Software	3		Cost for Performing Regression Testing	5
	Costs for Enabling the Use of Standards	3		Cost of Producing Documentation	2
	Costs for Applying Code Generation	1		Costs for Hiring Quality Personnel	2
	Costs for Lowering Programming Productivity	1		Costs for Improving the Quality Assurance Process	2
	Costs of Over-Engineering	1		Costs for Performing Glass-box Testing	1
	Costs for Applying Quality Function Deployment	1		Costs of Prototyping	1
	Costs for Improving System's Usability	1		Costs of Simulation	1
Failure	Necessary Rework (Debugging, Quality Improvement)	55		Costs for Performing System Testing	1
	Declining Market Positioning	17		Costs for Enabling Test Automation	1
	Additional Costs for Marketing	3		Costs for Performing Usability Testing	1
	Lost Opportunity Costs	3			
	Costs Caused by Damages	2			
	Costs by Investing on Failure Recovery Research	2			
	Costs for Hiring Quality Personnel	2			
	Costs Caused by Legal Damages	2			
	Costs Caused by Non-Operational Losses	1			
	Costs of Supporting of Multiple Versions	1			

3.3 Business Goals and Financial Benefits

Among the studies mentioned in Karg et al. [11], we have been able to identify only one study that mentions concrete benefits (at the business level) in software development. Although this study was considered, it was not particularly well-cited in literature or accepted as an authoritative source. Thus, we opted instead to adopt the set of business goals as established by Bass et al. [2], which is one of the most cited and recognized sources in the architecture community and was derived in practice through numerous industrial software development projects. An alternative source for identifying business goals and benefits, is the Pedigreed Attribute eLicitation Method (PALM) [38], which also helps to relate business goals with architecturally significant requirements.

Bass et al. propose the following business goals [2]:

- ***Growth and Continuity of the Organization*.** The possible contribution that a software system can have on the sustainability of the organization, focusing on its growth and continuity.
- ***Meeting Financial Objectives***. The extent to which a system can contribute to the revenue generated (if sold as standalone or a service) or saved by its use (if used for internal purposes).

- ***Meeting Personal Objectives***. Individuals have various goals associated with the construction of a system, including the increase of personal reputation, knowledge, and experience gains.
- ***Meeting Responsibility to Employees.*** This goal refers to issues related to employees involved either in development or in operation, such as ensuring their role in development teams, giving them opportunities to evolve professionally, or provide them a safe working environment.
- ***Meeting Responsibility to Society***. Some organizations see themselves as being in business to serve society; e.g., limiting resource usage, green computing, ethics, safety, open source issues, security, privacy.
- ***Meeting Responsibility to State.*** Organizations that develop government systems, are intended to meet responsibility to state (e.g., export controls, regulatory conformance, or supporting government initiatives).
- ***Meeting Responsibility to Shareholders.*** Goals complementary to the financial objectives' category. Liability protection and regulatory conformance are issues that meet the goals of this category.
- ***Managing Market Position***. This goal refers to topics related to the strategy used to increase or hold market share, various types of intellectual property protection, or the time-to-market.
- ***Improving Business Processes***. Through this goal stakeholders aim at improving business processes, which in turn can enable new markets, new products, or better customer support.
- ***Managing Quality and Reputation of Products***. This business goal is related to product quality from the perspective of customers, such as branding, recalls, types of potential users, quality of existing products, and testing support and strategies.
- ***Managing Change in Environmental Factors***. Since business context for a system might change, this goal intends to encourage the stakeholders to identify change scenarios in the business goals for a system.

4. ARCHITECTURAL DECISION-MAKING AS A FINANCIAL INVESTMENT

In this section, we present the proposed architectural decision-making approach, namely ***ADMIT (Architectural Decision-Making as a financial InvestmenT)***. The central idea of the approach is that each decision is treated as a financial investment; as aforementioned in Section 1, the scope of ADMIT is *cost-benefit estimation of individual design decisions*, leaving out other elements of a decision or relations between decisions. Thus, all parameters that affect an individual decision are monetized, so that they can be aggregated as total benefits or costs, with the latter comprising the costs of prevention, appraisal and failure. The proposed approach involves five steps, as presented in Figure 3. In the next paragraphs we elaborate on each of the steps of the approach. In addition, to make the steps more concrete, we demonstrate them through a running example that is executed with the help of the accompanying tool. The running example will be expanded further on the case study section; it concerns an IoT system, where a decision needs to be made to move part of a computation from the CPU to the GPU so as to save energy consumption and improve performance.

Figure 3: Architectural Decision-Making as a Financial Investment

Initialization. The decision maker (e.g., a software architect or project manager) first provides some general information on the project under study that could range from completely technical parameters, such as performance, maintainability, or reusability, to purely business ones, such as the competitiveness of the tool in the market, etc. As a first step, the user can either select the project to which the decision corresponds to, or create a new one. While

creating a project, some basic characteristics of the project are being initialized, such as the Lines of Code, Market Size, Market Share, etc.

Identification of a Decision. Next, a need for making an architectural decision (e.g., adding a new library, switching to a new framework) is considered. In this step, the user creates a new decision, or selects an already existing decision to revisit. In our running example, we revisit the decision to move part of the computations from the CPU to the GPU (see Figure 4).

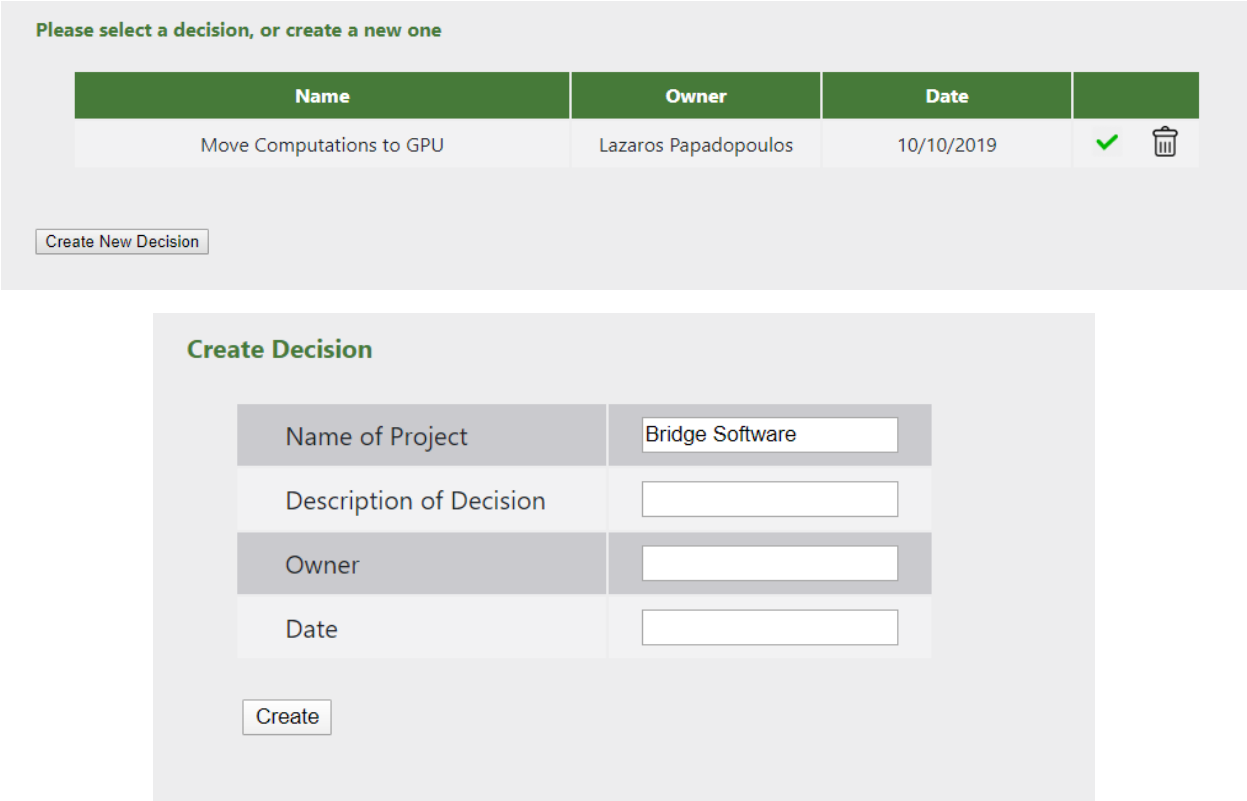

Figure 4: Identification of a Decision (Selection or Creation)

Attach Cost and Benefit Models to the Decision. In this step, the decision maker considers the costs and benefits that potentially arise from the decision. To facilitate the application of this step, the approach comes with some predefined cost and benefit models. As default cost models we have used the most frequent costs of Table 1; specifically, we considered all costs that were mentioned in at least 3 primary studies (see Section 3.2). As default benefit models, we used the business goals discussed in Section 3.3. We note that the number of business models is limited to three (compared to the eleven business goals in Section 3.3), since the rest could not be straightforwardly linked (in a generic way) to monetized benefits, directly gained from the outcome of the decision. For example, consider the goal "*Meeting Responsibility to Society*", and suppose a company that develops software that optimizes CO_2 emissions in cars. This company gets a better brand name, since they have contributed towards a 'greener' automotive, but this benefit can only be quantified by the increase in the market share that the company gets in other products, or new contracts. In such cases, we prompt the architect to use the default model on increased market share; however, the users of the approach can still use any other business goal by appropriately monetizing them.

Due to space limitations the complete list of default cost and benefit models is provided in Appendix B, which lists them along with an equation that shows the parameters of the models and the type of aggregation. The equations and

aggregation types are defined based on our perception and the definition of cost / benefit models in the original study. To ensure a common and sensible perception of the cost / benefit models that lead the development of acceptable equations, the authors performed a short focus group. The focus group was comprised of the authors themselves (one of them having financial background, and three of them experience in professional software development) and 3 individuals that are currently active software engineers / architects. The outcome of the focus group was a first validated version of the models, while the accuracy and the ease of instantiating these models (i.e., predicting the values of the parameters), is subject to further validation by our industrial case study. We note that the unit for all functions are currencies (e.g., $, or €). Finally, the fact that the number of default benefits is significantly lower than the number of default costs, is because benefits are more strongly related to specific decisions. For instance, an architectural refactoring that aims at reducing energy consumption, would lead to less energy costs for the users of the software. However, providing this very specialized benefit as a default one, might be confusing for the vast majority of decision-makers, who would not use it. One can observe from the table of Appendix B, that the provided models are either *time-framed* or paid/obtained *at-once*. These two types can (and must) be considered in the same model, in the sense that they are both valid for a decision. The default is to use a one-year period for the assessment of all time-framed costs/benefits so as to be able to combine them in a single cost-benefit model. If one has a different business model, one can timeframe the decision differently. The only restriction is that the same timeframe needs to be applied comprehensively to all equations. We emphasize that apart from the default cost-benefit models, the decision maker is free to develop custom ones that fit the purpose of a specific decision. If the default models are used, the decision maker can change the parameters and weight of each parameter in the corresponding equation.

Tool-wise, upon the creation of the decision, the decision-maker needs to select the cost and the benefit models that correspond to that decision. For this, there are two options: (a) create a new model (see Figure 5); or (b) select a default cost or benefit model (see Figure 6). While creating new cost / benefit models, the decision-maker first provides a name for the custom model (e.g., "*Cost of Migrating a Solution to the Cloud*") and then selects the type of equation (i.e., linear or product). Given the nature of the equation (e.g., $Y = a_0 + a_1x_1 + a_2x_2 + ... + a_nx_n$), the decision-maker declares the $x_1..x_n$ variables and their weight for aggregation, as well as the constant value (a_0). For instance, supposing that migration to cloud costs a standard amount of 1000$, plus 3$ per migrated line of code, the decision-maker needs to add two parameters: (a) the standard amount—*type*: constant, *value*: 1000 and (b) the lines of code to be migrated—*type*: parameter, *weight*: 3.

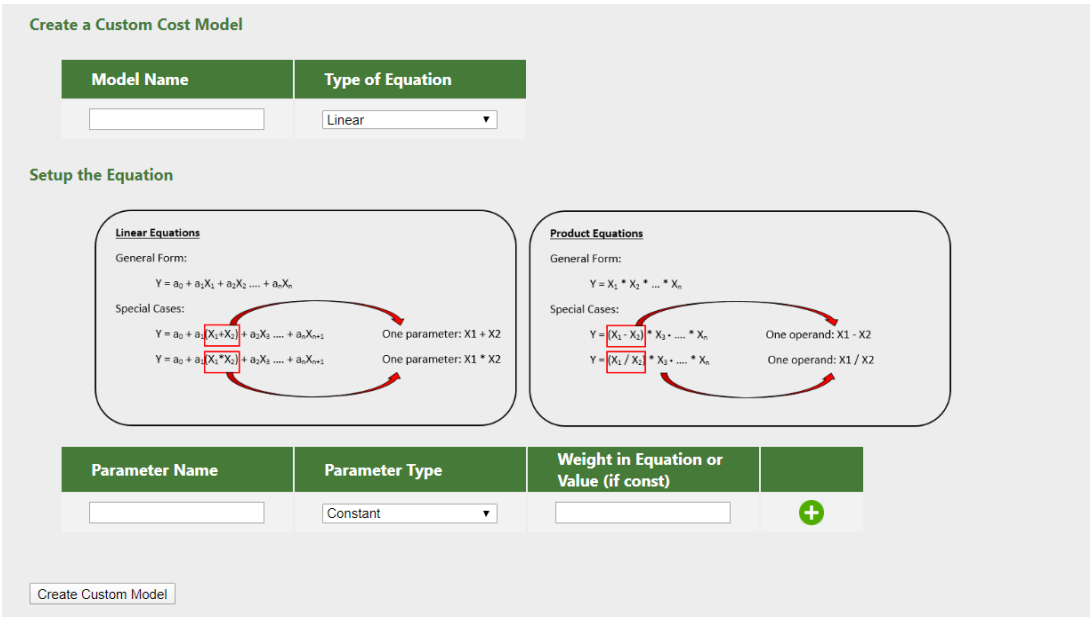

***Figure 5*:** Create a Custom Model

Select an existing cost model, or add a custom one

Type	Name	Equation
Cost Model	Acceptance and Integration Testing	Cost to Develop Test Cases + (Test Cases to be Executed * Time Needed to Execute Test * Testers Hourly Rate)
Cost Model	Code Inspection	(Number of Inspections per Month * 12) * Developers Hourly Rate * Number of Participants
Cost Model	Declining Market Positioning (e.g., poor quality, delay of release)	Lost Customers * Price
Cost Model	Design Reviews	(Number of Reviews per Month * 12) * Analysts Hourly Rate * Number of Participants
Cost Model	Documentation	Analyst Hourly Rate * Number of Specifications Required * Time Needed to Develop a Document
Cost Model	Hire Quality Personnel	New Employees * Yearly Cost of Employee
Cost Model	Increase Preventive Maintenance Effort	(Debugging in Hours * Developers Hourly Rate) + (Preventive Maintenance in Hours * Testers Hourly Rate)
Cost Model	Lost Opportunity Costs	Return on Best Foregone Option -1 * Return on Chosen Option
Cost Model	Metrics-Based QA	Cost to Retrieve Metrics + (Number of Metrics * Interpretation Time * QA Expert Hourly Rate)
Cost Model	Necessary Rework (e.g., Debugging, Quality Improvement)	Developers Hourly Rate * Needed Rework in Hours
Cost Model	Perform Meetings	(Meeting Duration in Minutes / 60) * (Meetings Per Month * 12) * Hourly Rate * Number of Participants
Cost Model	Project Planning	Additional Time for Planning * Manager Hourly Rate
Cost Model	Prototyping	Developers Hourly Rate * Number of Prototypes * Time To Build a Prototype
Cost Model	QA Tools	Cost of Acquisition + (Training Sessions in Hours * Instructor Hourly Rate)
Cost Model	Quality Audits	(Number of Audits per Month * 12) * External Expert Hourly Rate * Number of Participants
Cost Model	Regression Testing	Cost to Develop Test Cases + (Test Cases to be Executed * Time Needed to Execute Test * Testers Hourly Rate)
Cost Model	Reuse Proprietary Software	Cost of Software + (Training Sessions in Hours * Instructor Hourly Rate)
Cost Model	Unit Testing	Developers Hourly Rate * Number of Tests in Project * Time To Develop a Unit Test
Cost Model	Use of Standards	Number of Standards to be Applied + (Time Needed for Standard Compliance * Company Hourly Rate) + (Training Sessions in Hours * Instructor Hourly Rate)

Create Custom Cost Model

Figure 6: Select from Default Model

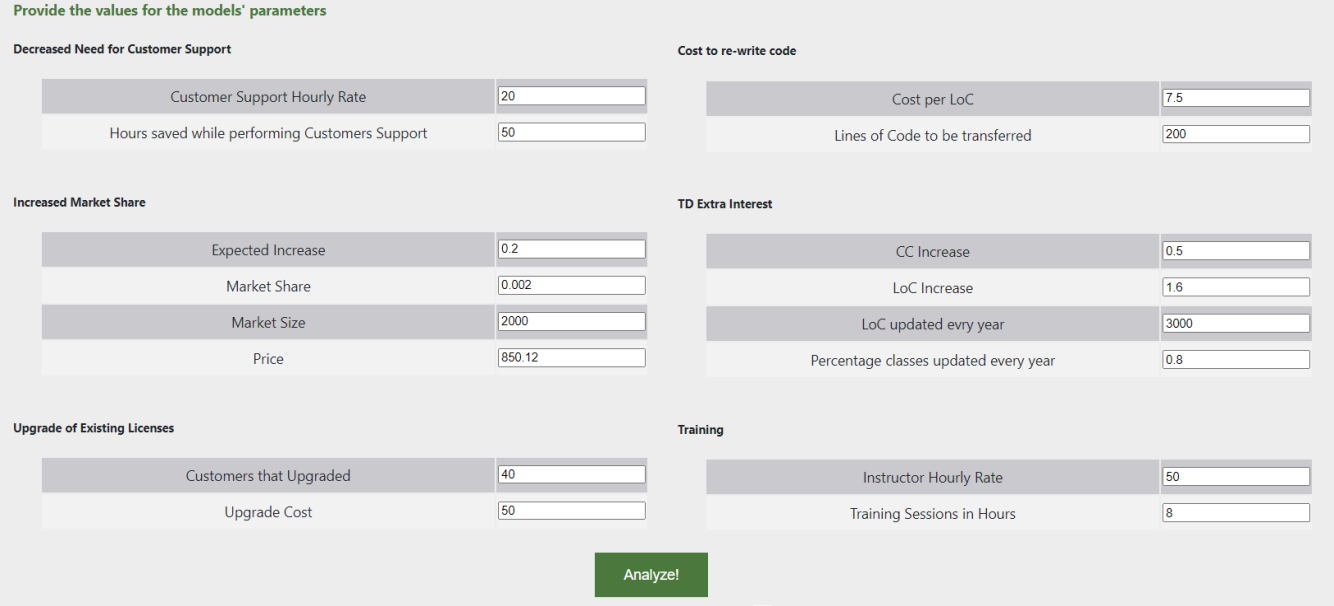

Figure 7: Instantiate Model

Instantiate the Models. In this step, the decision-maker can instantiate the models by providing the values to all parameters, so that the actual costs and benefits can be calculated. We note that since the values of the parameters are not necessarily single values, but ranges as well, the model is able to visualize the cost-benefit models, as functions of these values. In the tool, the decision-maker is provided with a list of all the parameters that affect a decision, and is asked to provide the expected values for the decision (see Figure 7). For instance, regarding the first parameter (i.e., "*Customer Support Hourly Rate*") of the first benefit model (i.e., "*Decreased Need for Customer Support*") the value has been set to 20$, as the company suggests that the monthly rate of the average personnel on this position is 2,900$ and the average man-month consists of 145 hours. We note that parameters (e.g., "*Testers Hourly Rate*") that are involved in more than one models (e.g., "*Cost for Performing Regression Testing*" and "*Cost for Performing Acceptance and Integration Testing*") need to be provided only once, and parameters that are among the default settings of the company (or project), e.g., lines of code, or amount of debt are automatically filled in by the tool.

Assessing the Decision. This final step enables the assessment of the decision (choose to either apply it or not), based on the *financial profit* of making the decision (total benefit – total cost). Upon the instantiation of the models, the decision maker is provided with the outcome of the cost-benefit analysis (see Figure 8).

Cost Model	Min	Max
Cost to re-write code	1,500.00$	1,500.00$
TD Extra Interest	1,920.00$	1,920.00$
Training	160.00$	400.00$

Benefit Model	Min	Max
Decreased Need for Customer Support	1,000.00$	1,000.00$
Increased Market Share	680.10$	680.10$
Upgrade of Existing Licenses	2,000.00$	2,000.00$

Instructor Hourly Rate	Total Benefit	Total Cost	Profit / Loss
20	3,680.10	3,580.00	100.10
23	3,680.10	3,604.00	76.10
26	3,680.10	3,628.00	52.10
29	3,680.10	3,652.00	28.10
32	3,680.10	3,676.00	4.10
35	3,680.10	3,700.00	-19.90
38	3,680.10	3,724.00	-43.90
41	3,680.10	3,748.00	-67.90
44	3,680.10	3,772.00	-91.90
47	3,680.10	3,796.00	-115.90
50	3,680.10	3,820.00	-139.90

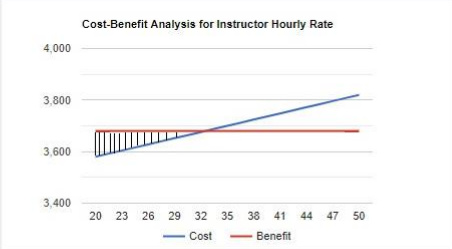

Figure 8: Cost Benefit Analysis

In the top part of the UI (Figure 8), the decision-maker can see an overview of the assessed cost and benefits for every model, given the input provided in Figure 7. Regarding the first benefit model of Figure 7 (i.e., "*Decreased Need for Customer Support*"), in the top-right part of the UI (see Figure 8), we can see that the benefit is 1,000$ (20$ per hour: first parameter of the model; and 50 hours saved: second parameter—see Figure 7). The lower part of Figure 8 appears only if the decision maker selects to tune the decision: If the assessment of the decision is not clear, the decision maker is provided with the option to select specific parameters and provide minimum and maximum values for them. For example, in our running case, the decision maker can observe that the decision to move the computations to GPU is largely non-profitable. We note that in the provided charts, the "*profit*" area is the one in which the "*red*" line is higher than the "*blue*" line (marked in the line chart). Therefore, he/she can check if there is a parameter, which if modified can constitute the decision profitable. In Figure 8, through the "*Select Parameter to Tune the Model*" feature, the user experiments on the cost of Training—based on the default cost model, the parameter of interest is "*Instructor Hourly Rate*". Through the line charts, the user understands that he/she should hire an instructor for training, paid at most 32$ per hour (point at which the blue line intersects with the red line), so that the decision is beneficial[1]. We note that the tuning of the parameters is not considered as a decision itself (i.e., the user does not decide to hire a lower rate instructor), but only as a way to explore how to make a decision beneficial.

5. CASE STUDY DESIGN

Case study is an observational empirical method that is used for retrieving empirical evidence in a real-life context [36]. To evaluate the proposed approach and tool to identify strengths and limitations, we conducted a case study. In this section we present the case study design, reported according to the template by Runeson et al. [28].

[1] We note that the graphical representations of Figure 8 include only linear models as an example; the tool can handle any other function

5.1 Research Objectives & Research Questions

The goal of this study in terms of the Goal-Question-Metric (GQM) approach [34] is formulated as follows: "***analyze*** the proposed decision-making approach and the corresponding tool ***for the purpose of*** evaluation ***with respect to*** their usefulness and usability ***from the point of view of*** software architects and managers, ***in the context of*** architectural decision making". Based on the abovementioned goal, we have derived three Research Questions (RQ):

RQ1*: What is the current status of architectural decision-making approaches and tools within the case study organization?*

Before performing the evaluation of the proposed approach and tool, we first need to understand the current way in which decisions are made. By evaluating the state of practice in architectural decision-making in the particular organization that we analyzed, we can understand if the proposed approach and tool treat existing limitations and retain the strong points. Thus, we investigate the involved stakeholders, the tools that are used, and the processes followed by the company. The benefits and drawbacks of using the already existing decision-making approaches are also discussed, to understand what already works well and what needs to be improved.

RQ2*: How is the proposed approach evaluated by practitioners in terms of usefulness and ease-of-usage?*

In this research question we explore the main aspects of the proposed approach, based on the specific context of the industrial partner. First, we explore the acceptance of the existing cost and benefit models that are introduced as default models in the approach. At the same time, we seek input for additional models that practitioners think that are missing from the default ones, aiming at evaluating the completeness of the provided default models. Next, we proceed with the evaluation of the approach, based on the two basic aspects of the Technology Acceptance Model (TAM): perceived usefulness and perceived ease-of-usage.

RQ3*: Does the developed decision-making tool meet the expectations of the practitioners?*

Having received input for the approach itself, in RQ_3, we focus on the offered tool support. Specifically, we investigate the usability (perceived ease-of-usage) of the tool and the extent to which it meets its functional requirements (perceived usefulness). Answering this question acts as a validation process for the tool and extends the body of knowledge on architectural decision-making. We note that, to answer RQ_2, we first need to demonstrate the approach to the subjects, so as to get a better understanding of its applicability and capabilities. Furthermore, to answer RQ_3, the subjects need use the tool; we do that by assigning two tasks to them and asking them to complete the tasks in a given time frame. Thus, we clearly separate the evaluation of the approach (RQ_2) from the tool itself (RQ_3).

5.2 Case and Subjects Selection

To answer the aforementioned questions, we performed an embedded multiple-case study in the software industry. The context of the case study is a Swedish SME, namely CNET, which is a software house specializing in research and innovation for Internet of Things technologies. The cases are two projects concerning the application of an IoT Box on two types of devices: PLCs in smart agriculture and bridges. IoT Box sensors gather vibrations, tensions, temperature etc., and send them to a cloud solution. The data is reported using the OGC (Open Geospatial Consortium) format and an open API allows smart phone apps to be created to render the data and process it for advanced analytics. The case study is embedded, in the sense that inside each case (IoT Box for smart agriculture and bridges), more than one unit of analysis have been studied. Specifically, the units of analysis correspond to the four participants of the case study: one project manager and one software/system architect per case. Some demographics of the participants are presented in Table 2 (the experience is measured in years). We note that since the scope of this study was to evaluate ADMIT per se, we looked for cases that: a) involve standalone decisions and not decisions

with multiple decisions and/or alternatives; b) do not use a systematic decision-making process that might introduce bias when being combined with ADMIT.

Table 2. Study Demographics

Participant	Role	Experience (in years)		
		Total	IoT Box	Decision-making
PM1	Project Manager	31	3	21
PM2	Project Manager	30	2	21
A1	Architect	24	2	18
A2	Architect	26	3	26

5.3 Data Collection

We collected qualitative data through different data collection methods, which are presented in Table 3 and are further discussed below. For all research questions, method triangulation has been applied to increase the validity of the findings. Method triangulation refers to the technique of mixing more than one method to gather data (such as interviews, observations, questionnaires, and document inspection, etc.) to answer a research question, so as to reduce bias, and raise confidence in the results.

Table 3. Data Collection Methods per Research Question

Collection Method	RQ_1	RQ_2	RQ_3
Interview	X		X
Focus Group	X	X	
Questionnaire		X	
Observations / Task Analysis			X

The case study was executed as a one-and-a-half-day workshop that was held in the premises of CNET, and was conducted by the second and the fourth author. The detailed schedule and organization of the workshop is presented in Appendix A—Section 1. The workshop started with a brief introduction of the goal of the workshop and the scope of the method, and an introduction of the problem.

The goal of RQ_1 is to understand the state-of-practice on architectural decision-making in the selected company. To achieve this goal, we first conducted interviews with the four participants. To guide the interview, we had a pre-planned guide (see Appendix A—Section 2). The interviews aimed at giving us some insight on the drawbacks of the current state-of-practice, so that we can check if these drawbacks are (at least partially) addressed by the proposed solution. Similarly, strengths of the current practices were identified and were used to check whether the developed approach would maintain such strengths. Next, we have performed a focus group, structured according to the guidelines provided by Kontio et al. [14]. During the **planning** of the focus groups we defined the goals: i.e., understand current practices in decision-making. Regarding the **design**, the focus group lasted for 45' and the participants were the same as in the interviews. While **conducting** the focus group the discussion was focused on the quality attributes that are central to architectural decision-making, by the involved technical and managerial parameters, the way to aggregate them, and the financial aspects of the decision-making—see Section 3 in Appendix A.

Regarding RQ_2 (evaluating the ADMIT method), we first demonstrate the application of the proposed cost-benefit analysis on a hypothetical decision to be applied in the IoT Box project of CNET. In particular, we illustrate the steps that would be needed to evaluate a decision to replace some CPU calculations with GPU calculations. This decision (as presented in Figure 9), would provide energy and performance benefits, but would cause the degradation of maintainability since the code would become more complex. We note that the demonstration scenario was

pre-determined and served only as an illustration of the process. To build a cost-benefit model, we have developed three indicative cost and three benefit models, as presented in Appendix A—Section 4, through a presentation and a detailed analysis of costs and benefits.

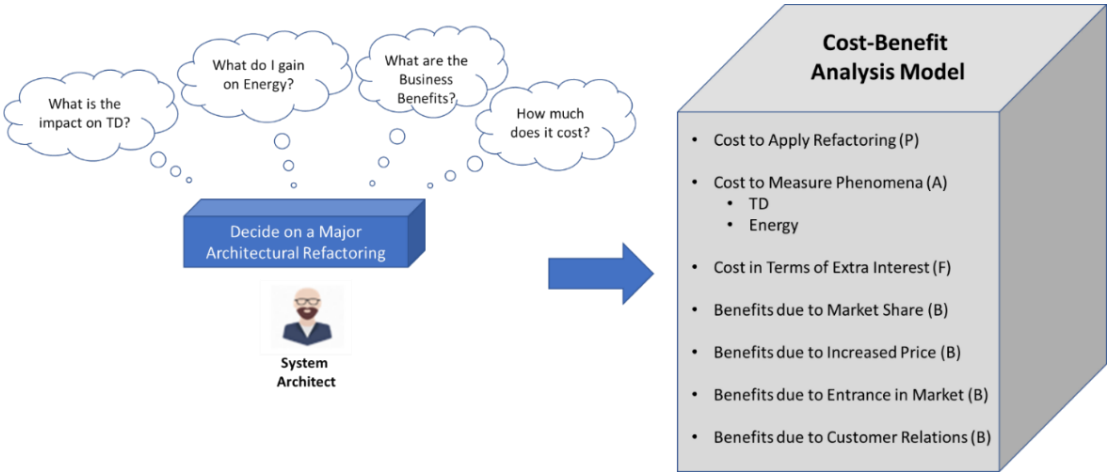

Figure 9. Workshop Demonstration Scenario

After providing the demonstration on how the aforementioned models are instantiated and how the cost-benefit analysis problem can be solved, there are two data collection sessions. The first one is a questionnaire, in which the default cost-benefit models of the approach (see Section 3.2) are being evaluated, using the structure presented in Figure 10. All questions are available in Appendix A—see Section 5. For example, consider the model of "*Exploit New Market Opportunities*". The monetization equation of this benefit (`NEW_MARKET_SIZE X NEW_MARKET_SHARE X PRICE`) is considered as accurate (given the accuracy of the values of the parameters); however, the practitioners might consider that a single decision (in isolation) is not adequate for opening new market opportunities (thus they have assigned a low frequency).

<<Cost/Benefit Model Name>>
To what extent do you think that this model is accurate? (Strongly Disagree → Strongly Agree) – 5 scales
With what frequency do you think that this model will be used? (Very Rarely → Very Frequently) – 5 scales
How easy is to assess the parameters needed to instantiate this model? (Very Difficult → Very Easy) – 5 scales

Figure 10. Questionnaire for RQ_2 (Structure)

Similarly to before, the focus group was designed based on the guidelines of Kontio [14]. In particular, during the **planning** of the focus group we defined the goals: (a) to discuss the details of the default cost/benefit models; (b) to explore the need for extending the set of default cost/benefit models; and (c) to evaluate the proposed approach. Regarding the **design**, the focus group lasted for 90' with the same participants. While **conducting** the first round of discussion (related to goal (a)), we used the structure presented in Figure 11—see Appendix A (Section 6, Part 1).

<<Cost/Benefit Model Name>>
What additional parameters would you need?
Would you use a different aggregation method?
Do you agree with the weighting?

Figure 11. Focus Group for RQ_2 – Part 1 (Structure)

Next, in the Part 2 of the focus group, we discussed goal (b), using the following questions as topics: "*What additional models would you need?*", "*Please define the model (parameters, aggregation method, weights)*", "*When would this model be used for?*", and "*What stakeholders would need to be involved to instantiate the model?*"—see Appendix A, Section 6, Part 2. Finally, regarding goal (c), the discussion was around the key expected improvements and liabilities of the method (e.g., in terms of usability, timeliness, accuracy, generalizability, etc.). The questions that have been used as drivers for the discussion in the last part of the focus group, providing an overview of its internal organization, are presented in Appendix A—see Section 6, Part 3.

Regarding RQ_3 (evaluating the supporting tool), we have organized three main sessions. In the first session, we used the task analysis method [19] by observing the participants performing two tasks on the tool, by plugging their laptops in a projector, and "recording" their rationale using the think-aloud-protocol (TAPS) [19]. Each participant was provided with a usage scenario and two tasks, as described in Appendix A—Section 5. Next, each participant, was interviewed with the goal of evaluating the functional requirements and the usability of the tool. To guide the interview, we have a pre-planned guide, as presented in Appendix A—Section 6. Finally, to assess the usability in a quantitative and more objective way, we used the System Usability Scale (SUS), which is a state-of-the-art method in the user interface design field [3]. SUS is reliable tool for measuring usability. It consists of a 10-item questionnaire with five response options for respondents; from Strongly agree to Strongly disagree. Originally created by Brooke [3], it allows UI/UX experts to evaluate a wide variety of products and services, including hardware, software, mobile devices, websites and applications. The items of evaluation are presented in Table 4. The participant's scores for each question are converted to a number, added together and then multiplied by 2.5 to convert the original scores of 0-40 to 0-100. Though the scores are 0-100, these are not percentages and should be considered only in terms of their percentile ranking. Based on the literature, SUS scores above 68 are considered above average and anything below 68 is below average [3].

Table 4. System Usability Scale

I think that I would like to use this system frequently	I thought this system was too inconsistent
I found the system unnecessarily complex	I felt very confident using the system
I thought the system was easy to use	I found the system very cumbersome to use
I think I would need the support of a technical person to be able to use this system	I would imagine that most people would learn to use this system very quickly
I found the various functions in this system were well integrated	I needed to learn a lot of things before I could get going with this system

5.4 Data Analysis

The dataset was inspected using lexical analysis for quantitative assessment and constant comparison for qualitative assessment. First, we transcribed all interview and focus group audio files, and we compiled them into a data set, including also the notes kept during the observation sessions. Then a lexical analysis took place: in particular, we have counted word frequency, and then searched for synonyms and removed irrelevant words. Then we coded the data set, i.e., categorized all pieces of text that were relevant to a particular theme of interest, and we grouped together similar codes, creating higher-level categories. The categories were created during the analysis process by both the first and the second author, and were discussed and grouped together through an iterative process in daily meetings. To visualize the outcome of this process word-clouds have been developed.

6. RESULTS

In this section we report the findings of this case study, organized by research question. Therefore, in Section 6.1, we present an overview of the state-of-practice in the company, accompanied by its perceived benefits and limitations. In Section 6.2, we focus on the evaluation of the approach, whereas in Section 6.3 on the tool evaluation.

6.1 State-of-the-Practice in Decision-making in CNET (RQ_1)

In this section we discuss the current state-of-practice in CNET for architectural decision-making. First, we describe two (recent) decisions that have been described by the participants, so as to set the context for the discussion. Next, we proceed with a summary of the current process, and a discussion of its limitations and advantages.

Context: As part of the case study the following two decisions were discussed: (a) PM1 and A2 referred to a decision that led to the development of a diagnostics' tool software (alternative: hardware update, or no action), which is able to self-check the IoT box, and could be extremely useful in cases of connectivity problems, especially for far remote bridges, which are not easily accessible (also due to weather conditions). In this case the quality properties that have been optimized were: reliability and robustness; and (b) PM2 and A1 referred to a decision related to caching and pre-processing data of edge devices, before sending them to the back-end (the cloud infrastructure). This decision was important for the scalability of the Smart Agriculture project, in which the option was either to increase the use of cloud resources (more costly for end-customers and not scalable), or re-write a large piece of the code.

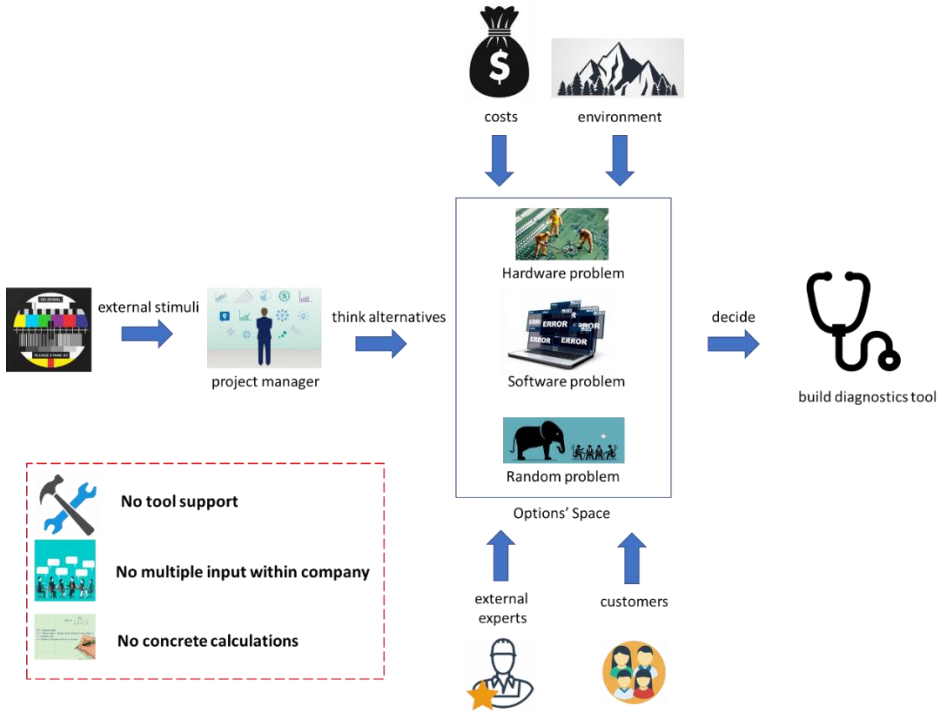

Figure 12. Smart Bridge Decision Panorama

The structure and the level of evidence in each decision was different, for each stakeholder, probably due to their level of seniority and experience. The first case (namely, the Smart Bridge System—see Figure 12) was discussed with PM1 and A2. The need to take a decision arose when the system (installed in a Bridge far away from Stockholm) stopped sending recordings. The project manager and the responsible architect had to decide if they would treat the problem as a hardware, a software, or a random problem. The initial thought of PM1 was described as follows: *"Based on my experience on similar problems in the past (by considering the higher cost of HW and the high*

probability that such a problem occurs again), I opted to go for treating the problem as a software problem, and build a diagnostic software to do the job". The decision was not taken only given the current situation, but by considering also the avoidance of future costs: *"Given the frequency of such connectivity failures, we believe that we have saved about 200-300K euros, by investing 2 weeks in the development of the diagnostics tool*". Discussing the same system, A2 pointed out that apart from cost saving other parameters have been considered, such as the environment ("*Most of them are placed far away from Stockholm and reaching them regularly for maintenance can become very costly. Thus, it is important to keep them robust*"), the opinion of customers and domain experts ("*to make the aforementioned decisions, we needed to talk a lot to customers, developers and experts in the development of IoT boxes, especially to get informed on the data consumption of such devices, and the reading capacity of the cloud services)*". In this decision the stakeholders have not used any tools, or made any concrete calculations.

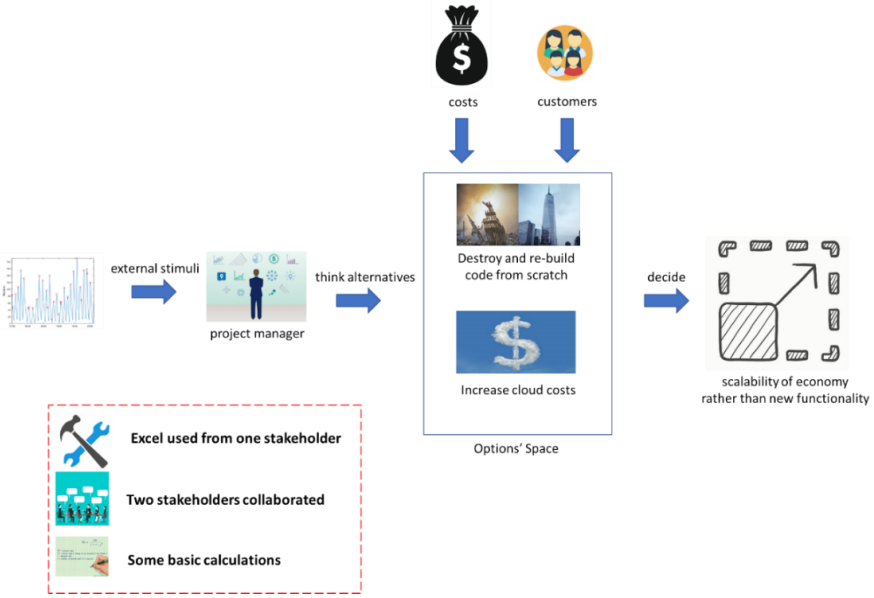

Figure 13. Smart Agriculture Decision Panorama

Regarding the second case, the Smart Agriculture project (see Figure 13), the two stakeholders exhibited a higher level of collaboration, since they were of different types: one more technical and the other more managerial. In the specific case of data caching, the decision started from A1, who noticed high peaks in the usage of cloud resources when adding farms in the IoT system. Similarly to the previous decision, the architect has also considered future versions of the system: "*The numbers seemed to be even more alerting by time-framing the problem in the upcoming year. Given the expected growth of the customers and the new installations in PLCs, it seemed to me that we need to make a decision on if we should re-write the software using caching libraries, or if we shall leave it for the future and increase the cost for end-customers".* A1, as a technical stakeholder person, could estimate the time required to re-write the code, but he was not able to assess the business perspective of this decision, so he needed to ask input from PM2: "*This decision was critical at that point, because from farming lots of data come in summer months, and if you do not react early, the system might 'crash' physically or economically in the peak of its usage. Thus, based on the estimates of A1, I decided to explore the business case more closely."* The collaboration between the stakeholders ended up in an excel file with the technical calculations (however, no specialized tool for decision-making has been employed), enhanced with the financial estimations that were made after talking to the customers. In the end the decision was to prioritize for enabling the scaling of the economy, rather than adding new functionalities; thus, customers were informed that in the upcoming sprints there will be no delivery of new functionalities, so as to

decrease the cost per installation. Along this decision, the stakeholders, driven by the need to collaborate, used excel for data exchange and simple calculations, but in general they also described their process as ad-hoc and conceptual rather than a structured approach, that could be generalized, reproduced and reused.

Analysis of Current Process: Based on both the interviews and the focus groups, the decision-makers of CNET suggested that the current-state-of-practice is an almost purely conceptual process, which is based on existing evidence and numbers that stem from various sources. These however are not systematically synthesized, but mostly based on their experience. A lexical analysis of the transcripts from interviews and the focus group have revealed the most common terms that the participants used, while describing the current state-of-practice and evaluating it. The results of the lexical analysis are presented in Table 5. From Table 5 it becomes evident that that they are already considering financial aspects of the decisions, such as the obtained *benefits* and primarily the *costs* required to apply the decision. Nevertheless, we need to note that the extensive use of these two terms might be subject to bias, due to the nature of the study. Additionally, the customers ("*what the customers need*") are crucial in the decision-making process, as well as *time*. We note that *time* is used under two perspectives by the participants: (a) as the time horizon for the decision, and (b) as the time required to perform an action. Finally, the participants highlighted the need for basing their decisions on *data / facts*, i.e. "*what the decision-maker really knows*", and not the personal opinion of the manager, and what "*he/she thinks on a given problem*".

Table 5. Terms Describing the Current State-of-Practice and its Evaluation

Terms	#	Terms	#
cost	93	support	34
customer	85	model	31
time	67	problem	26
need	57	functionality	25
data / fact	53	performance	23
information	51	issues	22
discussion	43	people	21
benefits	39	tool	21
quality	38	scaling	19
process	34	parameters	18

In a retrospective evaluation of the conceptual process that is used for decision-making in the previous examples, the involved stakeholders identified specific benefits and limitations. On the one hand, the main ***benefit of the existing approach*** is its *simplicity* and the fact that it *saves time*. All participants agreed that the used conceptual process *exploits the knowledge of the decision maker*. In particular, A2 claimed that having a knowledgeable manager poses a benefit, since he/she always decides based on what he/she really knows: *"It is good to base your decision on what you really know and what you have experience on. This brings more confidence to the final decision".* Concluding, the participants agreed that this approach is manager-centric and relies on how knowledgeable the decision maker is. Nevertheless, it was explicitly stated that before applying any decision, the decision was discussed among high-level managers, being presented and supported by the decision-maker.

On the other hand, ***limitations*** have also been identified. In particular, participants stated that the current approach:

- lacks *structure and control*. According to the stakeholders, this lack of structure and control can lead to less thorough thinking of the problem space, and eventually end-up with not identifying all possible alternatives, or evaluation criteria. These problems can lead to the approach not scaling well in larger problems.

- does not enable decision *reuse*. The reused parameters and rationale, are not documented in any way, and therefore any similar decision has to be assessed from scratch.
- is not easy to *involve various roles* and *support collaboration*. Using this conceptual process does not enable getting data from the most knowledgeable person in the company, as PM2 described: "*Inevitably we cannot know everything! At various points of decision-making you need information that you do not have. In such cases it is important to have stakeholders directly involved in the process rather than acting as a hub of collecting knowledge*".
- might *induce bias* from the perspective of the decision-maker. In particular, it was claimed that when presenting and supporting the decision to other managers, the decision maker is always positively biased in persuading others that he/she has selected the most fitting alternative.

6.2 Evaluation of the ADMIT Architectural Decision-Making Approach

In this section, we focus on the evaluation of the proposed approach for supporting architectural decision-making. We note that despite the fact that we have developed an approach and an accompanying tool, we decouple the evaluation of the two; that is why we did not present the tool to the participants of the case study, while evaluating the approach. The approach was evaluated under two perspectives: first, we evaluated the default cost and benefit models, which are central to the success of the approach; second, we evaluate the approach itself, and the extent to which it alleviates the limitations of the current decision-making approach in the company.

Evaluation of Default Cost and Benefit Models: The evaluation of the default models took place in two steps: first through a questionnaire that aimed at evaluating the *accuracy*, the anticipated frequency of actual usage (*applicability*), and the *ease* of applying the cost and benefit models (see questionnaire structure in Figure 10). The results are presented in Table 6. To aggregate the scores from the four practitioners, we added the value that they assigned (1: strongly disagree – 5: strongly agree / 1: very rarely – 5: very frequently, and 1: very difficult – 5: very easy). Thus, the total scores can vary from 4 to 20. In Table 6, the following notations have been used, similarly to a heatmap:

- with green cell shading, we highlight cost/benefit models that score in the range (16, 20], i.e., the majority of votes ranges from "*Strongly Agree*" to "*Very Strongly Agree*", "*Frequently*" to "*Very Frequently*", or "*Easy*" to "*Very* Easy".
- with light green cell shading, we highlight cost/benefit models that score in the range (12, 16], i.e., the majority of votes ranges from "*Neutral*" to "*Strongly Agree*", "*Neutral*" to "*Frequently*", or "*Neutral*" to "Easy".
- with grey cell shading, we highlight cost/benefit models that score in the range (8, 12], i.e., the majority of votes ranges from "*Disagree*" to "*Neutral*", "*Rarely*" to "*Neutral*", or "*Difficult*" to "*Neutral*".
- with red cell shading, we highlight cost/benefit models that score in the range (4, 8], i.e., the majority of votes is: "*Strongly Disagree*" to "*Disagree*", "*Very Rarely*" to "*Rarely*", or "*Very Difficult*" to "*Difficult*".

Table 6. Default Cost-Benefit Models Evaluation

Cost Model	Accuracy	Applicability	Ease
Costs for Hiring Quality Personnel	18	12	13
Costs for Performing Meetings	17	13	16
Costs of Training	12	13	14
Increase Preventive Maintenance Effort	14	14	13
Cost of Producing Documentation	14	14	12

Cost Model	Accuracy	Applicability	Ease
Increase Cost for Project Planning	17	16	17
Cost of Prototyping	13	15	15
Cost for Reusing Proprietary Software	13	15	18
Cost for Performing Code Inspection	17	13	17
Cost for Performing Design Reviews	17	14	18
Cost for Performing Unit Testing	16	16	12
Cost for Using Quality Assurance Tools	15	14	15
Cost for Performing Quality Audits	16	11	16
Cost for Performing Acceptance and Integration Testing	17	16	14
Cost for Performing Metrics-Based Quality Assurance	14	11	11
Cost for Performing Regression Testing	14	11	13
Necessary Rework due to Failures (e.g., Debugging, Quality Improvement)	16	13	13
Declining Market Positioning	13	12	8
Additional Costs for Marketing	11	12	11
Lost Opportunity Costs	15	12	7
Benefit Model	**Accuracy**	**Applicability**	**Ease**
Increased Market Share	15	11	8
Upgrade of Existing Licenses	18	15	15
Decreased Need for Customer Support	18	17	16

From the aforementioned table, we can observe that out of 23 models, only 1 seems to under-perform (grey or red shadings—below *Neutral*) in accuracy, 4 in terms of frequency of usage, and 5 in terms of ease of instantiation. On the contrary, 22 out of 23 models, score higher than *Neutral* in terms of accuracy, 19 out of 23 in terms of applicability, and 18 out of 23 in terms of ease of instantiation. The aforementioned findings are visualized in Figure 14, through a stacked frequency bar, for each evaluation parameter (i.e., accuracy, applicability, and ease of use). The results suggest, that cost models are accurate, applicable, and easy to instantiate. Among the examined evaluation parameters, the one achieving the lowest score (although still quite high) is the ease of instantiation. A possible interpretation for this outcome is that identifying the values for all parameters is an aspect that requires input from various stakeholders, and is quite subjective. For example, "*Additional Costs for Marketing*" requires specialized marketing knowledge, for which only one of the participants was aware of.

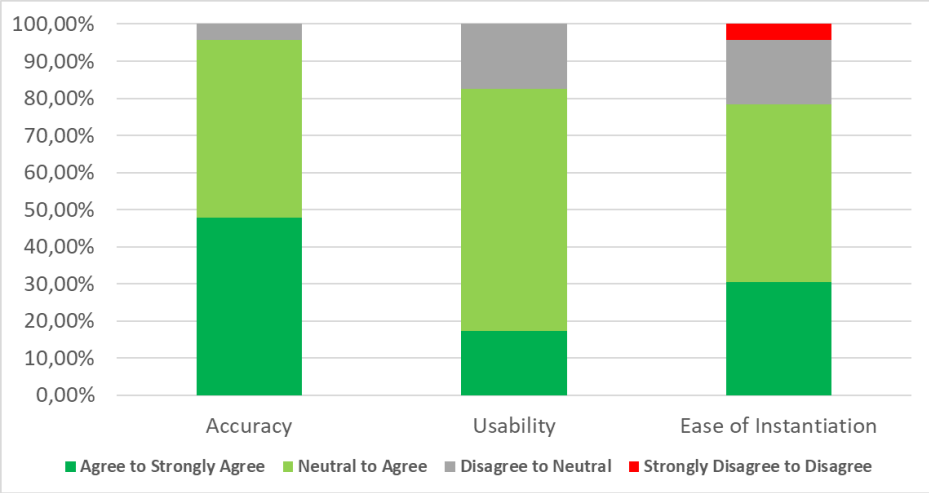

Figure 14. Frequency of Evaluations in terms of Accuracy, Applicability, and Ease of Instantiation

As a counter measure, to improve these values, during the focus group, we discussed these cost / benefit models, and identified ways to improve their quality. For example, "*Costs for Performing Meetings*" could be improved by entering different rates for all participants, i.e., naming a-priori the participants in meetings and retain their exact hourly rate from HR services. This would improve both the accuracy and the ease of instantiation. The same strategy can also be applied in "*Cost for Performing Metrics-Based Quality Assurance*". Regarding the most 'problematic' models, we have been advised to update "*Declining Market Positioning*", so as to take into account future sale, whereas in "*Additional Costs for Marketing*" to also consider indirect costs (such as investment material, marketing campaign material, etc.) and not only personnel cost. In terms of missing cost and benefit models that could be reusable in many projects, the participants highlighted the need for models that are related to cost for improving security, and costs for improving the user interface of a systems, such as usability testing, which they consider different from "traditional" testing.

Evaluation of the Proposed Approach: In general, the participants considered the financial view of the approach as beneficial. In particular, the participants suggested that the aggregation of various views under the same unit (i.e., currency) solves two main issues:

- ***improvement of communication within company and management***: Participants consider communication as a "*constant problem*", since "*everyone thinks that his job is the most difficult and important one*". So, the use of such an approach for aggregating different views can provide a "*global understanding*", a "*global point of view*", and "*set a common ground for discussion*" before reaching a decision. Also, setting a common ground is expected to "*bring balance between the various stakeholders*". Also, the participants agreed that "*money is a great unit, that everyone understands".* Therefore, misunderstandings and underestimation on the importance of parameters are avoided. The use of the "*breaking point is a nice feature*", since practitioners suggested that it is very interesting to "*play*" with the values of variables, so as to check multiple scenarios, without any cost.
- ***handling of various qualities and trade-offs***. The participants believe that the approach can help in "*conscious and objective decision-making*" for trade-offs, that it enables upfront thinking and helps in "*avoiding bias*". Using this approach can lead in understanding if "*it is worthy taking a decision*". Also, the approach enables finding a "*balance between the risk and cost of failure*", especially in cases when the opinion of various stakeholders (representing the different qualities) needs to be considered. However, even when us-

ing such a structured approach, they make some "*best guesses for the values of the parameters*". So, experienced people are still needed, since experience cannot be subsumed by a structured decision-making approach.

Additionally, the participants have not found the logic of the approach itself complex, despite the fact that identifying the values of some parameters might be difficult. On the contrary, they believe that "*problem complexity (which is large) is now distributed to many people (i.e., the various stakeholders)*". PM2 suggested that after reaching a "*maturity state in using the method*", you can reach a "*golden-level of detail, in which you achieve a perfect balance between bureaucracy and gains*". However, before reaching this level, it is important to build a culture among stakeholders that management "*shall not be central*", but it shall be "*distributed to knowledge holders*". Nevertheless, building this culture will be much easier with such a financial approach, since "*the use of money is helping in this direction, since people consider them important*". Finally, all participants agreed that applying the approach is very easy after modeling the problem.

Regarding the applicability of the approach in specific contexts and the required competencies to correctly apply the approach, the participants were also positive. By also considering the software development method, the participants believed that "*it perfectly fits agility*", since feedback can be given in iterations, models can be continuously updated, and "*architectural decisions or refactorings can be incorporated into sprints*". Furthermore, the participants highlighted that the distribution of decision-making responsibility, does not require an "*omniscient project manager or decision maker*", since each person could contribute the "*piece of knowledge*" that he/she knows best. Finally, the "*reuse of decision-making that the approach offers*" enables less experienced decision-makers to "*learn from the experience of others*". Apart from the aforementioned positive feedback, multiple useful suggestions and substantial feedback for improving the tool have been provided; two examples are given below. First, the participants agreed that a default setting should be provided to all stakeholders before providing their input, so that one does not change the parameters of others (e.g., the size of the company, the seniority and rate of individual employees should already be available before usage, the setup of meetings shall be constant, etc.). Second, the participants agreed that risk and time are important factors in this kind of decision-making, and calculations of the various factors should consider them. We note that the notion of risk as part of architectural decision-making has been extensively discussed by Poort and van Vliet [26].

6.3 Tool Evaluation

In Section 6.2, we evaluated the proposed approach, in isolation from the developed tool, so that the approach evaluation is not biased by possible problems of the tooling. In this section, we focus on the tool per se, aiming at identifying its current strengths, but also weaknesses that can drive its future developments. The tool evaluation session was divided into three main phases: first we observed the practitioners in performing two tasks in the tool, so as to get an indication of their ability to use the tool unassisted; second, we asked them to fill-in the SUS questionnaire; and third, we conducted some interviews to get qualitative data on their impression and suggestions for future developments. Below, we report the findings based on the two goals of this research questions: usefulness (based on the interviews) and ease of use (based on interviews, task observation and the SUS questionnaire).

Usefulness: The results of the evaluation of the anticipated functional requirements are summarized in Table 7. Specifically, in Table 7, we present the most commonly occurring keywords in the discussion of each functional requirement, and some quotes that back the claim. It is evident from the results that the usability of the tool can be improved, but the feedback is generally positive and many advantages of using the tool have been highlighted.

Table 7. Evaluation of Functional Requirements

FR	Quotes
Equation / Model Creation	"The use of the tool is straightforward" "The tool provides reuse in terms of equations and parameter, but this shall expand to values, and ranges of values" "Time is an important parameter for equations, we should have the ability to add it as such"
Visualization	"Quite easy to get the breaking point of the decision" "2D plots is a reasonable way to show the results, I would not expect to more complex visualization" "Ranges of values could be replaced with a scrollbar"
Parameter Tuning	"Observing the contribution of equations in the loss/profit is a very nice feature" "The tuning of parameters gives you the opportunity to experiment, even with fields outside your expertise"
Customization	"It would be nice if I could copy/paste models" "The fact that I can add a pessimistic and an optimistic view is a good way to customize the model instances"

As a tentative list of improvements, the participants have provided us with multiple suggestions that are currently under development in the tool. The proposed usability improvements are summarized below:

- decisions should be *exportable* in some common format, like *.pdf*, or *.xls* files
- the user should have the ability to *save values* for specific *parameters* that should be visible to all decisions of the same project, or company
- *save multiple results* for the same decision, so that you can compare in a visual way
- *handle communication between stakeholders* from within the tool, e.g., sending emails or private messages
- there should be a *details section* in the equations and results panels, so that you can add information / assumptions for values or parameters.

Ease of Use: Based on the observation session, we have been able to validate that the participants could correctly perform the tasks under consideration. All participants were able to follow the instructions and translate the instructions into tasks that need to be applied in the tool. Also, the way of thinking of developers confirmed their understanding of the approach, since researchers only had to intervene 3 times to correct the rationale of the participants. Such interventions were initiated by the practitioners, who explicitly mentioned that they had difficulty understanding something, and needed the help of the researchers to proceed. Since the interventions were different for each participant, no direct conclusion on the ease of use could be drawn; therefore, further details are omitted from the manuscript. We note, that all participants were able to build the models, instantiate them, and tune them, so as to experiment with the parameters and identify the best setup for future tasks. In terms of SUS, the participants rated the usability of the tool from 70% to 82%, which is considered sufficient for a research prototype tooling. The main negative points that stood out from the SUS questionnaire was the existence of some bugs, and the (expected) lack of confidence of the evaluators when using the tool for the first time. With respect to usability evaluation, we explored the effectiveness, the efficiency, and user satisfaction received from the tool. Regarding effectiveness, all participants were confident that they used the tool appropriately, so as to correctly complete the task. In particular,

the participants highlighted that they could very easily navigate from the description of the decision to the results, since the tool was simple to use. The participants, were especially satisfied with the tuning of parameters, since it provides more confidence in the decision-making. However, they noted that experienced users should have the opportunity to work only through the keyboard, without the need for the mouse, in order to be more effective. In terms of efficiency the stakeholders suggested that the task took reasonable time for completion, including the time to check the decision. However, the time required, would be increased if the models were not given (though, this is something unrelated to the tool, but mostly related to the approach). Finally, with respect to user satisfaction, the end-users seemed very satisfied from the tool, since they could imagine themselves using it in their day jobs: "*I can see how I could use the tool in my work, I can even now imagine some cost models for the storage decision. It would be easy and interesting to build some cost models for that with the tool*". Another participant, added a different parameter in the daily usage of the tool: "*I could do that. However, only for projects that run for many years*". In terms of complexity all participants were satisfied with the simplicity of the tool; one of the participants claimed that "*the problem is certainly complex, but the tool is very simple*". Therefore, they manifested that they would be able to use the tool without any technical support, or documentation: "*a try-and-error process seems to work well, because the tool is very simple and the functionalities are intuitive. If it also becomes robust, there will be no need for manual*".

7. DISCUSSION

In this section we discuss the main findings of the paper from three perspectives: (a) we compare the findings to existing literature; (b) we provide tentative interpretations of the main results; and (c) we provide useful implications for researcher and practitioners.

Comparison to Related Work: In this paper we explore the process of decision-making related to architectural decision-making, by: (a) exploring current practices in industrial software development—based on a case study in a single company, (b) propose an approach for architectural decision-making, accompanied by tool support; and (c) evaluate the proposed approach and tool in an industrial setting. Regarding industrial practices used for architectural decision support, our results from interviews and focus groups with industrial practitioners are aligned with related work. First, we confirmed the findings of Manteuffel et al. [22] that the majority of participants do not use a particular, systematic approach for decision-making. For the case of our study, the participants described the used approach as a conceptual process that is neither structured, nor documented. Second, the results of the study validated the claim of Poort and van Vliet [26] that supporting decision-making with financial data (e.g., cost or risk) can facilitate the communication with business stakeholders. Regarding the proposed approach, our work complies with this of Seaman et al. [29], who proposed to use the cost-benefit analysis method from economics, and integrate in the decision-making process the business perspective, as supported by de Almeida et al. [6]. Both aspects have been appreciated from the industrial stakeholders: on the one hand they suggested that using currency as a unit for presenting the results, is important to attract the attention of involved stakeholders, especially of managers; on the other hand, they suggested that customers and therefore business parameters are key aspects in decision-making. Finally, the proposed approach goes beyond related work, in the sense that: (a) it provides more detailed guidance on how the cost-benefit modeling will be performed, since it provides a set of reusable and highly-applicable (as suggested by our results) set of default models; (b) it provides a tool that guides the process of parameter selection, weighing, and the comparison of alternatives, as well the real-time tuning of the models; and (c) aggregates all outputs under a unified unit (i.e., a currency) that is well-accepted by industrial stakeholders.

Comparison of the Proposed Approach to the Existing Approach: When answering RQ_1, we listed both the strengths and the drawbacks of the existing approach in the case company. Our plan was to assess whether the pro-

posed approach is able to maintain these strengths and also address the drawbacks, at least partially. This assessment is summarized in Table 8, organized based on the pros and cons of the existing approach.

Table 8. Comparison of Proposed Approach to Existing Approach

	Existing Approach	Proposed Approach
Pros	Saves Time	The participants confirmed that the proposed approach and the tool were easy to understand, and the time required to apply the proposed approach is minimal, especially after the cost/benefit models have been designed. Nevertheless, all participants confirmed that the development of models is expected to be a time-consuming process, especially for the first decisions. It is expected that the time to develop models can be reduced by: (a) reusing decisions; (b) reusing parameters; and (c) further familiarization with the approach.
	Simplicity	The participants mentioned that despite the fact that the approached problem is a very complex one, the proposed approach is very simple and does not require any special training or manual.
	Exploits knowledge of decision-maker	The participants acknowledged that the decision-maker, no matter how knowledgeable he is, may not know every detail. So, it is more beneficial to combine their knowledge with data from the most accurate stakeholder.
Cons	Lack of structure and control	The participants suggested that the proposed approach provides a frame for setting up the mental process that they used, so as to be more structured and disciplined.
	Not enabling reusability	The participants claimed that the documentation provided by the approach and the imposed organization, constitute the decisions more reusable for future cases, either for revisiting them, or reusing "as-is".
	Not supporting collaboration	The participants highlighted that there is always a need to get input from various stakeholders. The collaboration opportunities that the approach and the tool provide, enable sharing the responsibility of making the decision with various stakeholders. This reduces the complexity of the problem.
	Inducing bias	The fact that the decision-making is now a collective responsibility, relieves the decision maker from the burden to stand up for their decision in company meetings. This, makes everyone more open to change opinions and support different suggestions.

Implications to Researchers and Practitioners: The findings of the study have enabled us to provide various implications for researchers and practitioners. On the one hand, ***decision-makers*** (i.e., project managers, architects, etc.) are advised to use tools, more structured processes, and rely on data and facts while attempting to reach their decisions. A structured way to achieve this goal is the use of the proposed approach and tool, which frames the decision space and has received positive feedback from industrial stakeholders, when assessing architectural decisions In addition to this, as a best practice, while making decisions, the experts have highlighted the involvement of various stakeholders (including customers), the monetization of decision parameters, and the consideration of time and risk

as important parameters in the decision-making. On the other hand, there are various interesting future work opportunities for ***researchers***. First, researchers are strongly encouraged to combine both technical and business perspectives while proposing approaches for architectural decision-making, since placing more emphasis on any of them, neglects important factors that need to be weighed. Second, based on the findings of the study, we highlight as an interesting extension of this study the following: (a) replicate the study in a larger context; (b) perform more research on benefit models, which until now seem to be neglected compared to cost models—e.g., use the Pedigreed Attribute eLicitation Method (PALM) [38] to extend our default set of business goals; (c) investigate appraisal and failure costs in more detail, since they seem to lag, compared to prevention cost models; and (d) explore opportunities for supporting the ease of instantiating these models by using automated techniques, based on data stored in company databases (both technical and business ones). Furthermore, we believe that an interesting future work direction to this study would be to extend ADMIT so as to handle inter-dependencies among decisions. Such an extension would need to rely on a network of decisions, whose outcomes could: (a) be used as inputs to other decision; (b) constrain other decisions—e.g., one alternative could be rejected, based on a previous decision; or (c) impose a decision—i.e., one alternative must be selected, based on a previous decision [17]. Finally, we would encourage researchers to investigate into more detail how time and risk (important parameters in decision-making, as claimed by the practitioners) could be used in such models: i.e., how they could modify and mix with models that are able to express all decisions in monetary terms. For instance, regarding time (since risk as a factor is already discussed in the literature [26]), including time as a parameter for the "*Increased Market Share*" model, could entail adding a new parameter "*years of expected sales*", which would stand for the timeframe in which the new market share would be applicable. This would not calculate the increase of revenue for one year, but for the lifetime of the product.

8. THREATS TO VALIDITY

Potential threats to validity of the conducted study concern construct, external, and internal validity and reliability threats. Since the goal of the study is not to establish any causal relationship, but only to provide an initial exploration, we believe that internal validity is not a main concern for this study's validity.

Construct validity reflects to what extent the phenomenon under study really represents what is investigated according to the research questions [28]. To mitigate construct validity threats, we established a research protocol to guide the case study, which was thoroughly reviewed by two experienced researchers in the domain of empirical studies. Additionally, during the data collection process we aimed at data and method triangulation to avoid a wrong interpretation of a single data source. Another threat is the fact that the tool and the approach have been evaluated separately, and without a long-term usage of the tool before the study; this has introduced both negative or positive bias. On the one hand (*negative bias*), the evaluation of the stakeholders was probably stricter, since the users were completely inexperienced with the tool, they have probably faced more usability issues, compared to an evaluation that would have been performed after some training or self-training period. Therefore, we believe that the presented results, correspond to the worst-case scenario of usage and evaluation. On the contrary (*positive bias*), the evaluation might have been positively biased by using only a demo session, in which unclear parts were explained by the researchers, making them easier to understand by the practitioners. Nevertheless, since practitioners had time to experiment alone with the tool/approach in RQ_3, we believe that possible problems while instantiating the cost/benefit models (i.e., providing values to the parameters) would have been identified and reported.

In terms of ***external validity*** (i.e., the generalizability of the findings derived from the sample [28]), it is difficult to claim that the same results would be derived in other companies. However, emphasizing on analytical generalization we can report on mitigation actions, which allow us to argue that the findings are representative for other cases with common characteristics (especially for RQ_2 and RQ_3). Specifically, the participants of the study were professional

software engineers with various years of experience in software development. Regarding RQ_1, however, the results might be difficult to generalize outside CNET, in the sense that decision-making and chain of responsibilities in SMEs may be very different from one company to another. Based on this, RQ_1 has been stated explicitly for the CNET context. Another threat to generalization is related to the evaluation of default cost/benefit models. In particular, we cannot claim that the evaluation results can be expanded to decisions involving other kind of cost/benefit models, since the default set has been developed based only on one study [11]. Nevertheless, we need to note that the study of Karg et al. [11] is a secondary study that provides an overview of the field of poor-quality cost modelling for several years, and therefore does not correspond to the personal view of some authors, but to the cumulative view of the completed domain. Finally, the general applicability of the approach is threatened by the fact that not all cost and benefits can be monetized; thus, ADMIT is applicable only to decisions that rely on cost and benefit models that can be assessed in currency forms.

The ***reliability*** of the case study concerns the trustworthiness of the collected data and the analysis performed, to ensure that same results can be reproduced [28]. We support the reliability of our study by creating a rigor case study protocol and interview guides, which were tested through pilots. To minimize potential reliability threats during the data collection process, we preferred to ask open-ended questions and we requested motivation for the provided answers. To assure the correct and unbiased data analysis, three researchers collaborated during the whole analysis phase. Finally, we have internally archived all collected data (both raw and coded), due to a non-disclosure agreement with our industrial partner. On the other hand, interview and focus group guidelines are openly available in Appendix A.

9. CONCLUSIONS

Architectural decision-making is a field that deserves investigation, in the sense that the number of decisions that need to be made is high in large software systems and can have substantial impact on software evolution. Software architects face various cases, in which they need to make a decision and effectively select the most fitting solution to an architectural problem. To make this process structured, controlled, and reusable, in this paper we propose a decision-making approach and an accompanying tool. The approach relies on studying both technical and business parameters of the decision and aggregates them by representing them in the form of cost and benefit models, that use currency as a unit for the calculations. Therefore, all parameters need to be monetized before "*entering*" the decision-making mechanism. The proposed approach has been evaluated in an industrial setting, through an embedded single-case study. The results of assessing the current practices in the industrial partner confirmed the lack of structured approaches and use of tools in the industry for making architectural decisions. The participants have positively evaluated the proposed approach, highlighting the importance of mixing various types of input (coming from different stakeholders) and aggregating them, using a well-accepted unit, such as money. Along evaluation, many positive aspects of the approach and tool have been highlighted, but also various points of improvement have been mined. Based on the findings of the study, various useful implications to researchers and practitioners have been provided.

ACKNOWLEDGMENTS

Work reported in this paper has received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No. 780572 (project: SDK4ED).

REFERENCES

[1] M. A. Babar and I. Gorton, "A tool for managing software architecture knowledge", *In 2nd Workshop on Sharing and Reusing Architectural Knowledge-Architecture, Rationale, and Design Intent (SHARK/ ADI'07: ICSE Workshops 2007)*, IEEE, pp. 11-11, 2007

[2] L. Bass, P. Clements, and R. Kazman, "Software Architecture in Practice", *Addison-Wesley Professional*, 2nd edition, 2003.

[3] J. Brooke, "SUS-A quick and dirty usability scale". *Usability evaluation in industry*, London, 189 (194), pp. 4-7. 1996.

[4] R. Capilla, F. Nava, J. Montes, C. Carrillo, "ADDSS: Architecture Design Decision Support System Tool", *In Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*, IEEE, pp. 487–488, 2008.

[5] R. Capilla, A. Jansen, A. Tang, P. Avgeriou, M. Ali Babar, "10 years of Software Architecture Knowledge Management: Practice and Future", *Journal of Systems and Software*, Elsevier, 116, pp. 191–205, 2016.

[6] R. R. de Almeida, U. Kulesza, C. Treude, and A. H. G. Lima, "Aligning Technical Debt Prioritization with Business Objectives: A Multiple-Case Study", *In 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME'18)*, IEEE, pp. 655-664, 2018

[7] A. Elizondo-Noriega, D. Güemes-Castorena, and M. Beruvides, "An analysis on the prevention, appraisal, and failure model in COQ: Convergences and contradictions", *IIE Annual Conference*, pp. 758-763, 2017.

[8] A.V. Feigenbaum, "Total Quality Control", *Harvard Business Review*, 34 (6), pp. 93-101, 1956.

[9] A.V. Feigenbaum, "Total Quality Control", *McGraw-Hill*, New York, 1961.

[10] J.M. Juran, "Quality Control Handbook", *McGraw-Hill*, 1st ed., New York, 1951.

[11] L. M. Karg, M. Grottke, and A. Beckhaus, "A systematic literature review of software quality cost research", *Journal of Systems and Software*, Elsevier, 84(3), pp. 415-427, 2011.

[12] R. Kazman, J. Asundi, and M. Klein, "Quantifying the costs and benefits of architectural decisions", *In Proceedings of the 23rd International Conference on Software Engineering (ICSE 2001)*, IEEE, pp. 297-306,2001

[13] S. Kim, and B. Nakhai, "The dynamics of quality costs in continuous improvement", *International Journal of Quality and Reliability Management*, Emerald, 25(8), pp.842-859, 2008

[14] J. Kontio, J. Bragge, L. Lehtola, "The focus group method as an empirical tool in software engineering", *In: Guide to Advanced Empirical Software Engineering*, Springer-Verlag, pp. 93-116, 2007

[15] P. Kruchten, P. Lago, H. Van Vliet, and T. Wolf, "Building up and exploiting architectural knowledge", *In 5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*, IEEE, pp. 291-292, 2005.

[16] P. Kruchten, "What do software architects really do?", *Journal of Systems and Software*, Elsevier, 81(12), pp. 2413-2416, 2008.

[17] P. Kruchten, P. Lago, and H. van Vliet, "Building up and reasoning about architectural knowledge", *In International Conference on the Quality of Software Architectures*, Springer, pp. 43-58, Berlin, Heidelberg, 2006.

[18] J. Lee, S. Kang, and C. K. Kim, "Software architecture evaluation methods based on cost benefit analysis and quantitative decision-making", *Empirical Software Engineering*, Springer, 14 (4), pp. 453-475, 2009.

[19] T. C. Lethbridge, S. E. Sim, and J. Singer, "Studying software engineers: Data collection techniques for software field studies", *Empirical software engineering*, 10(3), pp. 311-341, 2005.

[20] Z. Li, P. Liang, and P. Avgeriou, "Architectural debt management in value-oriented architecting". *In Economics-Driven Software Architecture*, Morgan Kaufmann, pp. 183-204, 2014.

[21] P. Liang, A. Jansen, and P. Avgeriou, "Collaborative software architecting through knowledge sharing", *In Collaborative Software Engineering*, Springer, pp. 343-367, Berlin, Heidelberg, 2010.

[22] C. Manteuffel, D. Tofan, P. Avgeriou, H. Koziolek, and T. Goldschmidt, "Decision architect–A decision documentation tool for industry". *Journal of Systems and Software*, Elsevier, 112, pp. 181-198, 2016.

[23] M. Nowak and C. Pautasso, "Team situational awareness and architectural decision-making with the software architecture warehouse", *In Proceedings of the 7th European Conference ECSA, Montpellier, France. In Volume 7957 of Lecture Notes in Computer Science.* Springer, pp.146–161, Berlin, Heidelberg, 2013.

[24] R. E. Peimbert-García, J. Limón-Robles, M.G. Beruvides, and D. García-Hernández, "An economic framework for total productive maintenance (TPM)". *In Proceedings of 62nd IIE Annual Conference and Expo*, Institute of Industrial and Systems Engineers (IISE), pp. 2760-2769, 2012.

[25] J.J. Plunkett and B.G. Dale, “A review of the literature on quality-related costs”, *International Journal of Quality & Reliability Management*, Emerald, 4(1), pp. 40-52, 1987.

[26] E. Poort and H. van Vliet, "RCDA: Architecting as a risk- and cost management discipline", *Journal of Systems and Software*, Elsevier, 85(9), pp.1995-2013, 2012.

[27] L.J. Porter and P. Rayner, “Quality costing for total quality management”. *International Journal of Production Economics*, Elsevier, 27(1), pp.69-81, 1992.

[28] P. Runeson, M. Host, A. Rainer, and B. Regnell, “Case study research in software engineering: Guidelines and examples”, *John Wiley & Sons*, 1st ed., 2012.

[29] C. Seaman, Y. Guo, N. Zazworka, F. Shull, C. Izurieta, Y Cai, A. Vetro, “Using technical debt data in decision-making: Potential decision approaches”, *3rd International Workshop on Managing Technical Debt (MTD’12)*, IEEE, 2012.

[30] D. Spencer, “Card Sorting: Designing Usable Categories”, *Rosenfeld Media*, 1st edition, April 2009.

[31] A. Tang, Y. Jin, and J. Han, “A rationale-based architecture model for design traceability and reasoning”, *Journal of Systems and Software*, Elsevier, 80(6), pp. 918-93, 2007.

[32] A. Tang, P. Avgeriou, A. Jansen, R. Capilla, M. Ali Babar, “A Comparative Study of Architecture Knowledge Management Tools”, *Journal of Systems and Software*, Elsevier, 83(3), pp. 352-370, 2010.

[33] U. van Heesch, P. Avgeriou, and R. Hilliard, A documentation framework for architecture decisions, Journal of Systems and Software, Volume 85, Issue 4, April 2012, Pages 795-820, Elsevier.

[34] R. van Solingen, V. Basili, G. Caldiera, and H. D Rombach, “Goal Question Metric (GQM) Approach”, *Encyclopedia of Software Engineering*, John Wiley & Sons, 2002.

[35] H. van Vliet, and A. Tang, “Decision-making in software architecture”, *Journal of Systems and Software*, Elsevier, 117, pp. 638-644, 2016.

[36] R. K. Yin, “Case Study Research and Applications: Design and Methods”, *Sage Publications*, 3rd Edition, 2003.

[37] C. Hofmeister, P. Kruchten, R. L. Nord, H. Obbink, A. Ran, and P. America, “A general model of software architecture design derived from five industrial approaches”, *Journal of Systems and Software*, Elsevier, 80 (1), January 2007

[38] P. Clements, L. Bass, “Relating Business Goals to Architecturally Significant Requirements for Software Systems”, CMU Journal, 2018, https://doi.org/10.1184/R1/6582992.v1