2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track

Software Quality Concerns in the Italian Bank Sector: the Emergence of a Meta-Quality Dimension

Daniel Russo, Paolo Ciancarini
Dept. of Computer Science, CINI & CNR-ISTC
University of Bologna
Bologna, Italy
{daniel.russo, paolo.ciancarini}@unibo.it

Tommaso Falasconi, Massimo Tomasi
Technology Strategy & Architecture
Deloitte Consulting
Milan, Italy
{tfalasconi, matomasi}@deloitte.it

***Abstract*—This paper reports on a Delphi-like study about the Italian banking IT sector's greatest concerns. A new research framework was developed to pursue this vertical study: domain and country specific, using a Mixed Methods approach. Data collection was drawn in four phases starting with a high level randomly stratified panel of 13 senior managers and then a target-panel of 124 carefully selected and well-informed domain experts. We have identified and dealt with 15 concerns about the present situation; they were discussed in a framework inspired by the ISO 25010 standard. After having mapped the concerns within the ISO standard, we identified the emergence of a new meta quality dimension which impacts both on software quality and architectural description. Our inductive outcome lets this meta dimension emerge connecting both ISO 25010 and ISO 42010 standards.**

***Keywords*-Software Quality, Information Systems, Delphi Study, Mixed Methods.**

I. Introduction

This research focuses on the identification of some relevant concerns in the Italian banking IT sector, investigating the opinions of several stakeholders i.e., banks and outsourcing companies, system integrators, software vendors, and consultants. In order to pursue the study we surveyed experts' opinions. For the panel composition we used the well-profiled contacts of an established IT consulting firm. At the end of the research process, we gathered results that we consider highly valuable for the following reasons. The final panel was composed of more than one hundred senior IT banking experts who are mostly in top managerial positions. Moreover, this is a vertical study of an IT banking sector in Italy, which has a number of peculiarities that are comparable with other countries. We report the emergence of a meta quality dimension, which links software quality and architectural description standards. Finally, we used an innovative research method, based on the epistemological paradigm of Mixed Methods research [1].

The two authors belonging to this industry are concerned about some growing issues that they are analyzing in the Italian IT banking sector. Most of their customers raised concerns about critical issues e.g., growing complexity of the information systems, unjustified layers and middleware stratification, difficult reverse engineering of their software applications, and costs explosion.

We did not find any existent relevant papers which addressed these issues, so we performed a vertical study on the Italian market. The banking sector in Europe, and more specifically in Italy, has always been characterized by national regulations and local software products that led to specific technological solutions built by national vendors. Currently, the most important Financial regulations are standardized within the European Union, e.g. MIFID, and the European Banking Authority is coordinating the national authorities, so the regulation differences among countries are decreasing, while the past national footprint remains significant in IT. In fact, we know from experience that the information systems of most banking institutions (especially tier 1 and banks) are composed of a collection of custom made applications and specialized software packages, even if in the software market you can find while Integrated Solutions that are able to cover all IT banking needs, without customization (COTS). Such Integrated Solutions have been partially adopted for a specific subset of banking products and processes or in small banking institutes (tier 3), mostly due to a lack of functionalities. The depicted scenario led to the growth of a typical Banking Information System characterized by a high number of applications and a spaghetti like architecture with a huge number of custom based software interfaces. Moreover, when dealing with the IT banking sector we know that country specificity is high, since it is governed by domestic rules which usually match the European ones but sometimes - possibly often - follow autonomous paths. This crucial topic for the economy needs more investigation, as suggested in [2].

Our results may be helpful in understanding in depth and with high internal and external validity (due to the Mixed Methods approach) the phenomenon of decreasing software quality. We are also interested in collecting experts' opinions as they are shared among the community at large.

We used a research methodology which merged inductive research (through Delphi) with a deductive one (survey-like), to provide a comprehensive analysis of the problem. In these terms, we integrated both forms of data collection within the same research. So, we embed one form of data within another to analyze different types of research questions [1].

We investigated the following research questions (RQ):

- RQ_1: Which are the major software-related concerns in

978-1-5386-2717-4/17 $31.00 © 2017 IEEE
DOI 10.1109/ICSE-SEIP.2017.10

the Italian IT banking sector?
- RQ_2: Are these concerns shared among the community of experts of the Italian IT banking sector?

Finally, we tried to map the concerns we found in the ISO 25010 standard, to assess them using an internationally recognized reference.

The structure of this paper is as follows. In Section II a short background explanation is given. We then present the research design and details of the Delphi–like study that was conducted, in Section III. This is followed by a presentation of the results of our study in Section IV. In Section V we discuss our findings, including the implications for research and practice, as well as the threats to validity of this study. Finally, in Section VI we conclude and outline future research.

II. Background and related work

Billions of dollars are spent in software projects since their success is an important competitive advantage of any organization [3]. Software quality is the most important success factor of information systems – "Software quality can determine the success or failure of a software product in today's competitive market" [4]. Analysis of the software engineering literature highlights the importance of product quality "in use" for industries which strategically exploit software functions. The available literature also largely supports the view that a critical success factor for most of the initiatives in a domain like the IT sector is the information available to guide and support the management of software quality efforts [5].

Kan and Basili [6] refer to *data-based decision making*, an approach that can be used for both software project and quality management. They warn against the collection of excessive and arbitrarily developed metric data. They emphasize the importance of developing focused, accurate and useful measures or metrics, which are based on specific managerial models. The Goal/Question/Metric (GQM) model links metrics and measures with the operational goals of the organization. The goals of the firm are more clearly defined through their translation into a set of *quantifiable* questions. The questions then drive a *specific set of metrics and data for collection and provide a framework for their interpretation* [6]. While GQM is extremely flexible and adaptable to a variety of situations, the most important reference for software quality is the ISO 25010 standard on software products quality and software in use quality.

III. Research Design

As researchers, we obviously have our epistemological bias, which usually remains hidden or implicit, even if they deeply influence our research [7]. Since we approached this research in an innovative way, we clearly decide to use *Pragmatism*. It derives from the work of Peirce, James, Mead, and Dewey, and arises out of actions, situations, and consequences rather than antecedent conditions (as in postpositivism) [8]. It focuses on the research problem, rather than the method. This is the philosophical ground for the Mixed Methods approach.

The pragmatic view does not commit to any system of philosophy and reality. Like Mixed Methods, both quantitative and qualitative assumptions are used. This is related to the view that the world has an absolute unity. Truth is what works at the time, it is not based in a duality between reality independent of the mind or within the mind [1]. Therefore, in Mixed Methods research, investigators use both quantitative and qualitative data because they work to provide the best understanding of a research problem [1].

This paper reports the results of a Delphi-like study modeled on the Delphi methodology (qualitative) and survey (quantitative). In order to put in evidence the threat factors we used a phenomenological approach [9]. Moreover, the survey is a semi-structured one so, beside closed questions based on an hybridized-Likert scale, experts are asked to express their open opinions about any single question. Thus, the survey itself is mixed quantitative and qualitative.

Generally speaking, empirical software engineering is developing new research designs according to its research questions, which may cross the threshold of traditional borders of the discipline due to the pervasiveness of software [10], [11]. The Delphi method is becoming a popular tool in the software engineering discipline [12], even though it is still not well known. On the other hands, Likert–based surveys are a popular quantitative research method in software engineering [13]. We first present a brief discussion of the Delphi method, since it is the underpinning method of this research. Then we discuss how the Delphi panel was selected, and provide details of the Delphi process i.e., threats validation and the survey one. Finally, we discuss the outcomes.

A. The Delphi-like Method

The Delphi method has proven a popular tool in Software Engineering [12] and more in general in information systems research [14], [15]. It allows to capitalize the experiences of the expert panel in identifying key risk concerns in the IT banking sector, identifying the most important factors by continuous feedbacks. The objective is to explore valuable informations through a structured process of knowledge collection from a panel of experts with controlled opinion feedbacks [16]. The process consists of a series of rounds in which each expert communicates his opinion through a structured form i.e., questionnaire, structured interview, collected by the researcher. It is an inductive data–driven approach, ideal for explanatory studies for which little empirical evidence is available [17]. Using a step–wise methodology the research question is narrowed down, from a multitude of issues, to a bunch of a few consensus-based factors [18].

After gathering experts' concerns on the Italian IT banking sector we consolidated it into 15 factors (RQ_1) with a Delphi style methodology. Then, we collected opinions and evaluations about the factors with a "target-panel" of 124 profiled experts (RQ_2). The administration of the whole process is represented in Figure 1.

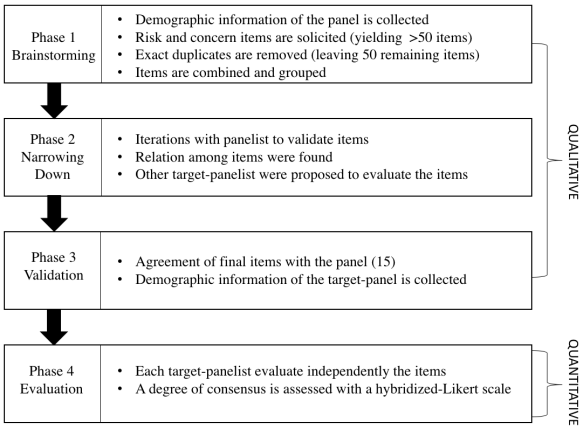

Fig. 1. Research administration process

		#	%
Company	Consultants	7	25%
	Bank	6	21%
	System Integrator	6	21%
	Outsourcing	5	18%
	SW Vendors	4	14%
	Total	28	100%
Experience	>10	4	31%
	>20	8	62%
	>30	1	8%
	Total	13	100%
Role	CEO/CIO	6	29%
	Chief Data Officer	5	24%
	Appl. Maint. Group Exp.	4	19%
	IT Architect	4	19%
	Maintenance Manager	2	10%
	Sales	0	0%
	Total	21	100%

TABLE I
PANEL COMPOSITION

B. Selection of Delphi Panelists

We are aware that the selection of panelists is the most critical aspect of Delphi, especially for validity threats. A lousy panel selection may compromise the whole qualitative research. Thus, we mitigated this risk using an established private dataset of an affirmed consultant firm; moreover the use of a double panel using Mixed Methods enforces the validation of both the questions and the answers. However, it still remains *the most important yet most neglected aspect of the Delphi method* [19]. Methodological literature agrees upon the fact that the choice of experts is the single most difficult factor in panel selection [20].

Hence, our greatest efforts were devoted to the selection of the first and then the second panel. Since we intended to perform a vertical study, country and sector specific, we put a lot of attention to the choice of experts invited invited to participate on the panels. We started from a privileged position. We used an expert pool of an established IT consulting firm, specialized in the banking sector, which works with all main banking groups. So, we were able to address personally, highly qualified experts (from the Italian IT banking sector). This makes our panel highly reliable and representative.

The first panel was chosen using a stratified random sampling. Strata were defined upfront, to define the sample population. Companies and roles were chosen to have a fair representation of the IT banking sector. The first panel did not follow the population representation task regarding experience, since we addressed explicitly senior experts, in agreement with [20]. Also the target-panel was chosen with a stratified random sampling, but inside a larger pool. We asked panelists to give their opinion about the sector composition (companies and roles) and also the name of other experts. Therefore, we adjusted the sample population representation along with our research journey, following a suggestion found in [1]. However, also for the target-panel we looked searched for more senior opinions, adapting the research method to our research purposes [21].

In Table I we describe the demographic composition of the first panel and in Table II that of the target-panel. We profiled our panels in great detail, much more than traditional Delphi studies. The guarantee of anonymity we gave to all experts allowed a very open and truthful discussion.

We took into consideration three dimensions: (i) years of experience, (ii) sector in which the experts worked, and (iii) relevant roles they served in their careers. Obviously, sector and role may be more than one per expert, since it is normal to change job during any career path. In detail, these are the dimensions and sub-dimensions surveyed:

Experience. We divided the experts into four groups: less than 5 years, between 5 and 10 years, between 10 and 20 years, between 20 and 30 years, and more than 30 years of experience. The greatest majority of our experts has more than 20 years of experience in the sector. Thus, we consider the opinions of our panel of high value.

Companies. We profiled also different company types, involved in the IT banking sector. Clearly, most experts displayed an internal experience within a *bank*. The second relevant companies are *system integrators*, since a high degree of customization is needed for bank products. *Consultants* and *Outsourcing companies* are also very important actors for the IT banking sector, since most work is outsourced to external workers or companies. *Software vendors* deliver software packages products for banks. As stated before, software packages are commonly used by small/medium sized banking institutes to either address specific needs (e.g. loans, deposit or transaction accounts) or support common processes (e.g. accounts payable and receivables). According to our experience, the composition of both the panel and the target-panel represent a trustful representation and distribution of companies in the Italian IT banking sector.

Role. (*CEO/CIO*) The Chief Executive Officer is the top manager of software integrator or vendor companies, while the Chief Information Officer is a board member, in charge of the

		#	%
Company	Bank	72	35%
	System Integrator	41	20%
	SW Vendors	34	17%
	Consultants	33	16%
	Outsourcing	23	11%
	Total	203	100%
Experience	<5	8	6%
	>5	1	1%
	>10	23	19%
	>20	72	58%
	>30	20	16%
	Total	124	100%
Role	Appl. Maint. Group Exp.	59	39%
	CEO/CIO	32	21%
	IT Architect	18	12%
	Chief Data Officer	18	12%
	Maintenance Manager	13	9%
	Sales	8	5%
	Other	2	1%
	Total	150	100%

TABLE II
TARGET-PANEL COMPOSITION

information system of the bank. The *Maintenance Manager* is in charge of the development and maintenance of IT banking systems. The *Application Maintenance Group Expert* is responsible for complex groups of IT banking applications. The expert in charge of the IT banking architecture is the *IT Architect*. *Sales* is a senior manager of the sales department of a software vendor or system integrator. Finally, the *Chief Data Officer* is an expert in industry standards and methodologies (e.g., IEEE, ISO, ITIL, DAMA) and in charge of the bank's data governance. Also the distribution of roles within both the panel and target-panel is representative of the Italian IT banking sector, at our best professional knowledge.

The methods we used to compose the two panels varied slightly. For example, we took into consideration for the target-panel the role of sales managers, not considered before. We had feedbacks that their role is also critical for the IT banking sector, so we adjusted it along the research process (following [1]). Senior experts were chosen for the first panel, with at least 10 years of experience within the sector. In the target-panel we wanted a wider and trustworthy representation of the sector, so we included also junior people in the target-panel. However, about two-thirds of the target-panel had more than 20 years of experience.

Most of our panelists and target-panelists experienced more than one position in more than one company. We did not find it meaningful to show just e.g., their last job, or the longest one. Experts expressed their opinion according to their general knowledge of the domain, which was gained through different professional experiences. According to our view, this enriches the research, since experts were able to judge the factors from different, highly qualified, points of views, within the same sector, in the same country. With respect to the financial year 2015, panelist represented banks which have an aggregated net worth of 82% and revenues of 95% of the national sector. For banking information systems, we mean all systems which supports banking operations in different market segments (e.g., retail, corporate) and different products/services (e.g., deposits, payments) and ERPs (e.g., risk management, communications). Concerns among banks are quite similar, since architectural patterns are shared by European banking regulations, or Local Standards as *Abilab*.

C. Data Collection and Analysis

The Delphi–like study took over a year for the four phases (see Figure 1), which are described next. According to the Mixed Methods approach, this research is composed of qualitative and quantitative approach. To develop the qualitative part we spent the first three phases, to identify the items. Then, we used a survey-based approach to validate the panel's items by a wider panel (target-panel).

The qualitative research started in October 2014 and lasted until April 2015. The survey was a little bit shorter, from June 2015 up to November 2015.

Phase 1: Brainstorming. After collecting the demographic information from our pool, and composing the panel, we conducted a brainstorming round to elicit as many concerns as possible. This phase was helpful to broadly understand the problem and seek factors. It was an unstructured process, where more than 50 items were solicited. Afterwards, exact duplicates were removed, leaving 50 items. Finally, items were combined and grouped by the authors.

Phase 2: Narrowing Down. In the second phase, further iterations involved the panel to validate the items. Relations among items and their grouping were discussed, also in an unstructured way. Experts were asked to discuss the grouping made by the authors of this paper and to redefine it, if needed. Items were represented in sticky notes on a blackboard. This helped the discussion over the items and their relations. Afterwards, authors wrote down the items list, in the terms discussed by the panel. Finally, the panelists discussed about the target-panel composition and also proposed other experts to invite.

Phase 3: Validation. The goal of the third phase was to validate the items. The panel considered more than 30 factors but it came finally to an agreement about the 15 final factors. Since we wanted to keep the whole process rather phenomenological, we did not use any statistical technique. Typical Delphi studies ask panelists to rank factors according to their importance and then aggregate it through Kendall's coefficient of concordance (W) to assess the degree of consensus among the panelists themselves [18]. We had a narrow group of highly qualified experts and managed them as a working group. The introduction of statistical methods for the validation of the final items was not grounded in our phenomenological approach. We wanted to grasp the essence of their experiences as described by the participants [9]. At the end of this phase, each panelist agreed on the final items and full consensus was reached, even though there were some little differences. For instance, the panelists did not always agree on the degree of consensus (i.e., strongly agree or just

agree). However, the baseline (i.e., agree) was always met for each item. This is probably due to the fact that both brainstorming and narrowing down phases involved a lot of personal confrontation. Therefore, during the validation phase, there was already a consolidated opinion about the items. Finally, some demographic information about the target-panel was collected.

Phase 4: Evaluation. The last phase concerned the quantitative inquiry approach, to evaluate the factors by a larger stratified sample group (target-panel). We prepared an online survey with the factors and made personal invitations to experts. Target-panelists could start the survey only after they inserted their personal credentials, given by e-mail or phone. To have control over the research and the randomized stratified sampling, no answer was anonymous. We used an hybridized-Likert scale for the factors evaluation. The aim was to compute semantic differentials (i.e., "Strongly Agree", "Agree", "Disagree", "Strongly Disagree") to define the level of agreement, typical of Likert scales [22]. We hybridized the Likert scale with even bipolar values (negative and positive), symmetric to 0, without an average effect. To stress possible differences, we gave higher values to both extremes, also to avoid an average effect. So, we assigned the following values: "Strongly Agree"= 3, "Agree"= 1, "Disagree"= -1, "Strongly Disagree"= -3. The idea was to highlight different semantic values of the two extremes and suggest equidistance between the center point of the scale and the two extremes.

IV. Results of the Delphi–like Study

The Delphi–like study resulted in a set of 15 factors presented in Table III along with the target-panel evaluation. All concerns were shared, at least, by 70% of the target-panelists with a different intensity degree (measured with the hybridized-Likert scale). Each factor is summarized followed by a short discussion where panelists' and target-panelists' opinions are directly quoted. For the sake of readability, only in this section, we use the terms panelist, target-panelist, informants and experts as synonyms.

Factor 1: Module interfaces complexity.

A Banking Information System is characterized by a high number of modules which are highly coupled. This increases the number of interfaces and their complexity. One panelist mentioned the spaghetti like architecture stating that *"we are experiencing a growing complexity in delivering new projects due to past implementation of point-to-point architectures delivered in the last years"*. Integration costs and higher complexity compared to a green field makes the business case less effective. Even worse, another one said that *"sometimes the interfaces to be updated are so complex that an ad hoc middleware is required"*. The lack of knowledge seems to be another root cause since a third panelist said *"I believe that the lack of knowledge about application functionalities led to code duplication over the years"*.

Factor 2: Interfaces architecture complexity.

This second factor is a direct consequence of the first one. Module interfaces complexity led to a typical anti-pattern [23].

#	Question	"Agree" and "Strong Agree"	Average score
1	Software modules interfaces are characterized by a high level of complexity and represent an important part of each module in terms of number of objects and LOC.	96%	1,95
2	Interfaces among modules are developed in a stratified way in time. This brought to a complex architecture hardly manageable and not future-proof.	96%	2,47
3	Software quality for custom development is decreasing in the last years (especially Cobol / CJCS / DB2).	77%	1,15
4	SW Maintenance & enhancements evolution costs and time of information systems are increasing due to the (i) stratification of software, (ii) poor documentation and (iii) low quality of the source code.	82%	1,58
5	Low software quality depends on increasing pressure for enhancements evolutions in shorter time with lower budget.	79%	1,57
6	Low software quality depends on a poor level of functional and technical analysis and detail.	79%	1,00
7	System analysis is hindered by an inadequate documentation and database.	87%	1,70
8	It is difficult to build and maintain effective documentation because of low budget and time shortening.	84%	1,46
9	New software packages have more functionalities than in the past but their increased complexity leads to difficult evolution management.	83%	1,58
10	Software packages are poorly documented for effective maintenance and evolution.	82%	1,30
11	Software packages documentation main gap is due to a poor description of the data managed by the system (not only the record layout, but also the fiscal and logical data model, the data dictionary, etc.).	77%	1,09
12	"Application Maintenance" contracts do not improve the software documentation.	77%	1,15
13	Non-Italian software applications are of higher quality but are not *per se* more maintainable.	76%	1,29
14	Italian software applications have more functionalities and lower quality but are not less maintainable.	75%	0,92
15	Software quality can not be reliably measured through tools and methodologies	70%	0,93

TABLE III
Factors and results of the Delphi-like study

According to a panelist, *"stratified software interfaces affects old developed applications; only using Service Oriented Architectures we took advantage of the strong benefits related to an integrated architecture"*. Old layered software is a remarkable problem as stated by another panelist *"there is a well known problem related to platform software, which were developed years ago and never replaced. Only a tactical update with a short run perspective"* was carried out. Continuous update of old layered software seems an attractive and affordable way, but long-term sustainability is questioned. Most concerns were related to the maintainability of such architecture. No refactoring solution was seriously taken into consideration,

due to cost. However, this short-term view did not decrease costs because the it is very probable that the system will stop working properly in a medium period and the replacement cost could be very high. Moreover, this leads also to unexpected problems which usually rise with such complex systems. A third panelist said *"we decided to replace the client data module and their interfaces, since we reached a point where we were unable to manage the evolution required by the business"*. Due the high degree of coupling of these modules (Factor 1), proper reverse engineering is required.

The next factors will show how the lack of documentation hinders reverse engineering. All these elements make it difficult to improve the banks information system.

Factor 3: Custom software quality.

Apparently, the quality of custom software applications is decreasing. Moreover, modules developed with old programming languages like COBOL, which are still widely adopted in banking, while there is a lack of young experts because such languages are not included in the current formal IT education programs. A panelist illustrated that *"the number of developers able to develop in COBOL are rapidly decreasing due to retirement. New developers are not skilled enough with COBOL and other mainframe languages because they are focused on the newer languages like Java"*. The interaction among old and new coding paradigms is another point raised by the panelists. One said that *"software quality is getting worse because we developed using a stack paradigm, adding new software layers on old software. Unfortunately this paradigm prevents the use of new technologies"*. Furthermore, the decrease of quality *"is perceived also in all other used programming languages"* according to other panelists. This may be also depend on the current training of developers. According to one expert *"several developers we hired did not go through a formal computer science education"*. This may be due to the high request on the job market of software developers who however get low salaries. So, skilled developers have usually many job offers and tend to choose the most profitable one.

Factor 4: Increase of maintenance costs.

Some factors have a direct impact on maintenance costs. The overall architectural complexity, the decreasing software quality and incomplete documentation are the most important drivers of high maintenance costs and time. As declared by a panelist, *"during the last six years our software modules have been impacted by a deep reengineering project and the most important effort was related to building new documentation"*. Another reason of the growing costs seems to be linked to skills: a panelist pointed out poor competences as primary cause of frequent module rebuilds that cause an increase in the application complexity. This is related to the use of different technologies *"it often happens that instead of updating the software, new applications are built on it. The coexistence of different technologies is an important cause of high maintenance costs"*. According to one panelist *"most costs are related to continuous regulatory changes requested i.e. by the ECB"*. This is, apparently, another element of stratification and architectural complexity.

Factor 5: Quality vs. Time & Budget.

The whole panel agreed unanimously that there is a direct relationship among quality and time and budget. One expert stated that *"a relevant cause of poor software quality is related to time constraints, these aspects have impact on quality"*. More time and budget to develop and evolve software properly would increase quality and decrease maintenance costs. In fact, *"an already well-written code could be evolved with low budget, while a stratified software which has been poorly written makes updating difficult, increasing costs and time"*. This is a chain-effect, one said that *"poor software quality is related also to software stratification due to low investments and the obsolescence of the information system. Poor implementation of software engineering methodologies due to low budget magnifies this crucial issue day after day"*. Another element which emerged is the relationship between the organization structure of banks and its impact on the information system. One panelist stated that *"it is of high relevance the organization structure of the customer to define the proper information system"*. Apparently, this element impacts on the quality of the system itself.

Factor 6: Quality vs. System analysis.

Even though the design phase is perceived as the most important up-front activity, it is poorly implemented. One panelist stated that *"it is necessary to invest in this phase, to get benefits within the whole life-cycle"*. However, *"it is poorly carried out, due to shrinking budget and time"*, said another informant. The problem may depend on the fact that often the role of IT departments is not perceived as highly critical by top management. So, *"there is not an adequate collaboration between business functions and the IT departments"*. Therefore, system analysis activity is often skipped because it is hardly tangible. Stakeholders are not willing to invest in some activities were they do not see an immediate return. Clearly, this leads to long term problems, as identified before. Another element is the that the formalization of the business requirements and a complete and effective vision of the information system is difficult. In this regard, one panelist affirmed that *"customers usually give poor and not coordinated requirements, leading to silos-like solutions instead of a fully integrated development"*.

Factor 7: System analysis vs. Documentation.

Inadequate documentation impacts on the system analysis and so on software quality. One expert stated that *"documentation is inadequate to the scope, being or too technical or too business-like with poor information abut the system"*. Moreover, a wrong interpretation of Agile methodologies results into poor documentation, in fact *"along with the stratification problem the misleading interpretation of Agile led to a poor and ineffective documentation"*. It is important to underline that the lack of data models documentation and metadata definition *"leads to an inadequate and dangerous system analysis"*.

Factor 8: Documentation vs. Time & Budget.

Time and budget constraints have a direct impact on software documentation. Due to low budget for new developments

and urgency for new applications, documentation is the first element which is skipped A panelist said that is impossible to keep documentation aligned with software both for the frequency of software update and the lack of methodologies used to develop and manage software. According to another panelist, *"constraints on project costs and time causes poor or no documentation. In fact, the documentation is usually delivered only if you have enough budget"*. Due to budget limitation, often banks prefer to skip documentation if they are offered a discount on the application cost. Also time plays an important role. Often, there is no time for documentation or, even worse, it is perceived as a waste of time. A panelist explained that *"budget constraints clearly impact on documentation, since we are not able to justify the budget required to keep documentation aligned. The only affordable way is to insert control points on the software development process (SDLC) to define the least amount of information necessary for maintenance"*. In this regard, documentation tools appear to play an important role. Another panelist declared that *"we can limit the problems that we have in software documentation thanks to the adoption of new generation tools (thanks i.e. to metadata) and the Agile approach"*.

Factor 9: New packages functionalities vs. Complexity.

For the reasons analyzed before, the demand for more functionalities rose in the last years, along with their complexity. Moreover, software vendors do not always apply industry standards. One panelist explained that *"new software packages requires methodologies and standards that only few big vendors can adopt. In this situation when we want to customize those packages we must rely on those big vendors but with a low degree of control and the danger of lock in"*. Some experts also said that *"recent packages are too complex and have lower quality than in the past"*. This factor explains the relationship between the market trend i.e., more functionalities with architectural complexity (factor 2) and dependency on vendors (factors 10-12).

Factor 10: Packages vs. Documentation.

The lack of documentation for software packages is perceived as a *"commercial strategy of suppliers to lock-in customers"*. This appears natural, since *"the development is often given to software houses"*. For one expert *"documentation is always lacking"* and it is not uncommon to *"buy packages without any kind of technical and functional documentation"*. Another reason may be the fact that *"suppliers tend to hide technical documentation as IPR protection strategy, delivering only the functional documentation"*. However, as another expert said, *"this problem needs to be tackled with a good Service Level Agreement"*, according to an expert.

Factor 11: Packages documentation vs. System analysis.

The lack of documentation in packages impacts directly on the logical data model and quality controls. As stated by a panelist *"information about data is one of the biggest problems, as well as the role that data plays on business lines"*. Moreover, *"process logic is needed to achieve a correct level of documentation. Just data are not enough"*. Also this factor suffers from low budget and scarce time.

Factor 12: *Application & Maintenance* contracts vs. Documentation.

Application & Maintenance (AM) contracts are set to outsource the development and maintenance, to decrease internal costs. Typically, they do not provide an adequate documentation. Therefore, when the supplier is changed, system evolution becomes rather difficult. Lock-in situations are very common since *"suppliers want to defend their know-how to maintain their competitive advantages"*. Like factor 10, *"a good Service Level Agreement is key to overcome problems"*. However, what happens is that SLA are not respected properly, to get higher discounts on services.

Factor 13: Non-Italian applications vs. Quality & Maintainability.

According to the panel, there is a difference between Italian and non-Italian software products, which is partially a concern. Apparently, non-Italian applications are more maintainable but have less functionalities. The reason seems to be that *"non-Italian applications are less flexible than Italian ones because they implement simpler functions and not because they are written or designed better"*. One panelist stated that for a well known, specific software application *"a low level of customization usually means lower license and maintenance costs"*.

Factor 14: Italian applications vs. Quality & Maintainability.

Regarding Italian applications, they appear to have more functionalities but incur in higher maintenance costs. For one panelist the reason seems to be that *"Italian applications are really rich of functionalities due regulatory requirements defined by the Italian Banking Authority"*. One expert clearly expressed a specific concern, stating that *"software applications should comply with international standards. Before the acquisition each customer has to test this compliance. In reality this never happens, and if it were the case most would fail. The only exception are applications delivered to NASA, US Air Force and so on, but at which costs?"*.

Factor 15: Measurement of software quality.

Losing the control over the system quality is a concern. First of all *"poor use of software engineering methodologies is the first killer of quality"*, according to one panelist. According to another expert *"even though methodologies are well known and some tools are available, they are not implemented within the software development process"*. The discussion about tools was quite interesting. *"Tools do exist but are extremely expensive and not suited for small banks"* stated one panelist. For another informant *"a tool suited for us do not exist on the market place, so we built it by ourselves"*. Finally, an expert very frankly explained that *"tools and methodologies are well known, however neither customers nor suppliers use them since none is willing to pay for them"*.

V. Discussion

The picture which emerges from our research is rather worrisome. The information systems of Italian banks appear to be characterized by a highly complex and stratified architecture,

with a sinking quality. Moreover, with respect to the past, experts' opinions let to emerge gloomy scenarios, due to an increase of functionalities set both by newer regulations and the current "digital transformation". The highly specificity of any bank business led to a large use of custom applications and software packages which in turn led to a hardly manageable architecture with an unjustified stratification of software modules and interfaces. Integrated solutions products, which may assure a more standardized quality, are rarely used because they do not fit all the banks requirements.

Cost cutting had a direct impact on architectural aspects (both *ex-ante* planning and *ex-post* analysis/reverse engineering), documentation and data modeling. These aspects are directly related to quality, since they directly impact the stakeholders of the software systems. However, traditional quality models, appears to be insufficient to explain these outcomes.

Even though this is a vertical study focused on a specific sector and country, we can assume that several issues are shared with other countries (especially EU ones). Cost-cutting needs are the same at least for EU and US banks due to the market low interest rates level. This reduced sensibly bank's margins and profitability, leading to generalized internal cost-cutting. Development was influenced by such business goal by delivering with lower budget (and time due to fast regulatory changes). Since custom applications as software packages implemented in Italian banks are comparable with other Eurozone's banks, this is a common feature.

The effort of this research was to map the identified characteristics within state of the art standards. Most of the 15 factors we identified are included in the Product Quality Model of the standard ISO 25010:2011, as shown in Table IV. To map our factors to sub-characteristic we used a Delphi approach derived from software cost estimation [24]. Each author elaborated autonomously the categorization. After that phase, we met personally and took for granted the unanimous cases. All other cases where also just one of the author was in disagreement were discussed. Since the factors which came out from the panel where rather cross-cutting along different quality characteristics, we assigned some factors on more than one characteristic. We choose not be restrictive in our categorization since the aim was to see if the standard is sufficiently comprehensive for the identified factors. Final factor mapping as described in Table IV is the unanimous results of the iterations of all authors. This reflects both perspectives from industry and university.

Although we mapped all factors, still some dimensions remain unexpressed. We found that the dimensions of experts' concerns are not all considered within the standard. For example, the documentation cost aspect seems to have been neglected in the standard. Moreover, in the literature there are no systematic methods or models to measure cost [25].

Apparently, a *meta dimension* of quality emerged. Some aspects like the relationship between cost and quality are not part of the standard (both as Quality in Use and Product Quality), even though it has a direct impact on software quality.

Characteristics	Sub-characteristics	Factors
Functional suitability	Functional completeness	6, 7, 12
	Functional correctness	6, 7, 12, 15
	Functional appropriateness	3, 6, 7
Performance efficiency	Time behavior	5
	Resource utilization	5
	Capacity	
Compatibility	Co-existence	5
	Interoperability	5
Usability	Appropriateness recognizability	7
	Learnability	8
	Operability	3, 9
	User error protection	
	User interface aesthetics	
	Accessibility	
Reliability	Maturity	8, 15
	Availability	
	Fault tolerance	7
	Recoverability	
Security	Confidentiality	
	Integrity	2, 7, 15
	Non-repudiation	15
	Accountability	2, 7, 15
	Authenticity	
Maintainability	Modularity	1, 2, 10
	Reusability	4, 8, 9, 10, 13 14
	Analysability	4, 5, 6, 8, 9, 10, 11, 12
	Modifiability	2, 4, 5, 8, 10, 11, 12
	Testability	4, 5, 8, 10
Portability	Adaptability	4, 8, 9, 10, 13, 14
	Installability	10, 13, 14
	Replaceability	4, 6, 8, 9, 10, 11, 12, 13, 14

TABLE IV
FACTOR MAPPING ACCORDING TO ISO 25010

Experts stressed in more than one factor this emergent dimension. Shrinking budgets and lack of development time impact directly on both architecture and software quality, according to our panelists. The nearest concept of the ISO standard within the Quality in Use is *efficiency*, intended as "resources spent in relation to the accuracy and completeness with which users achieve goals". But our experts never expressed an efficiency dimension. They reported about the relationship between time & budget on software and architecture.

Business goals have a direct impact on the information system. If the business (i.e., the bank's executive board) has as its goal the IT cost-cutting in order to invest this savings in other departments or to be more profitable the next quarter is a clear business goal. Short term-view decisions or even not-taken decisions are business goals which influence both software and architecture. Apparently, non-IT executives are willing to pursue short-term goals, increasing both the total cost of ownership and the technical debt. We did not collect non-IT experts' opinions in our study, so we can not explain this phenomenon. Like the panelists remarked, this led to low maintainability & evolution capabilities as also to some architectural anti-patterns. Several quality dimensions are influenced by such business goals, like quality, maintenance & evolution and architecture. It is like this *meta* dimension of quality, impact on all the others e.g., Use, Data and Product, Architectural Concerns, which have been already formalized within industry standards (ISO 25010 and ISO 42010).

Therefore, we do not induce (intended as the outcome of a qualitative research) another quality dimension. We induce

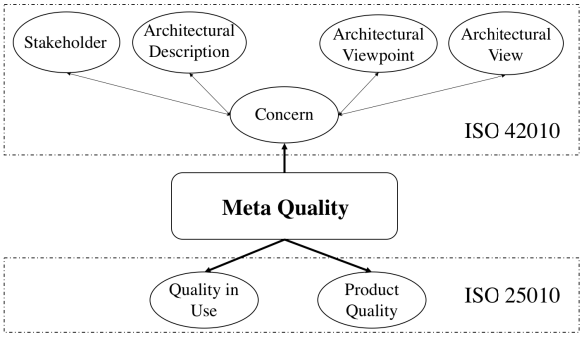

Fig. 2. Meta Quality dimension

instead a new *meta quality dimension* of software.

Architectural aspects were continuously stressed by our informants, letting emerge a relationship between business goals with both software quality and architecture. Apparently, this meta quality dimension impacts architectural concerns as intended by the standard of architectural description. According to ISO 42010 "a concern pertains to any influence on a system in its environment, including developmental, technological, business, operational, organizational, political, economic, legal, regulatory, ecological and social influences". In our study, business goals define architectural concerns, on which software architectures are built on. For example, the business goal to save on development influences the "system in its environment".

In our view this new dimension origins from the business goals and structure of an organization, following Conway's law [26]. It changes along with the organization's evolution. This is the main reason that it is definable as a *meta* dimension, since it changes over time. But, at the same time it impacts on quality, in all its sub-dimensions.

In literature this idea is not new, even though it was never observed with inductive approaches [27]. A meta quality model which defines the structure of operationalized quality models which bridges the gap between concrete measurements and abstract quality aspects (i.e., ISO 25010) has recently been proposed, starting from the standard incompleteness [27]. Such approaches narrow down quality dimensions in quality metrics. What differs mostly from this approaches is the incremental versus the holistic view of this attribute. We do not think a narrow definition of quality sub-characteristics into quality metrics solves the issue of system's quality. However, there has been maturing in literature the idea of the inadequacy of ISO standard's quality characteristics.

Our contribution is that of a typical qualitative, inductive study to have reported the emergence of a new dimension which has not been observed before in a real-world context. The proposed model is represented in Figure 2 and it links both ISO 25010 standard regarding software quality, as also ISO 42010 about architecture description.

A. Threats to Validity

Threats to validity were our biggest concerns during this study, since we used a new inquiry methodology through Mixed Methods. Therefore, we use both qualitative and quantitative validity paradigms. To analyze the qualitative dimension we adopt: credibility, transferability, dependability, and confirmability, according to [28]. While, for the quantitative dimension we use traditional statistical conclusion, internal, construct, and external validity by [29]. Even though validity dimension of qualitative and quantitative research are different, we aggregated them but discussed them separately due to epistemological reasons. The research was homogeneous and both qualitative and quantitative dimensions gained trustworthiness one from another.

Credibility & Internal. Factors identified are all credible. We identified them through Delphi-like process which last about one year. Panelist were sector experts, with a daily exposure to the researched concerns. The whole panel had the chance to discuss, refine and group the elicited items, through different phases. None of the experts argued that any of the factors should be excluded. Random stratified assignment of the research subjects, were designed to maximize internal validity. Representativeness of the sample is high for two reasons. Experts were chosen among a highly qualified pool of an established IT consulting firm of the bank sector. Strata were first assumed by the author's experience and than integrated into the panel. Moreover, the guarantee of anonymity given to both panelists and target-panelists allowed a unbiased and frank discussion. This increased the credibility of the study, since experts were able to answer and express openly their knowledge.

Transferability & External. The Mixed Methods approach aims to explore and build new theory (induction) and also to validate it (deduction). The degree to which the results of qualitative research are transferable to other settings may be interesting. However, we were strictly focused on a very specific sector in one country. We believe that most concerns are shared among other countries, for the reasons before described. This study helps the community to identify the most important factors which threaten the IT banking sector.

Dependability & Construct. The authors of this research are well aware of the context. Experts were carefully chosen through a stratified randomized sampling, using the important pool of an established IT consulting firm, specialized in the banking sector. We described the three dimensions (experience, company and role) and sub-dimensions, as also stratification of the samples. At the end of the qualitative research, a very wide target-panel of 124 carefully selected experts through stratified randomized sampling were asked to evaluate the panel's outcome. The outcome was a substantial agreement, on all factors with a degree of agreement above 70% for all factors, with high average scores. However, even though the presented 15 concerns were highly shared among target-panelists, this does not mean that these are the most relevant for them. Moreover, some elements may have been

lost during the translation from Italian into English. We remark that the factors were elaborated by a high level panel of expert in a three-phase round. The aim of the quantitative part was to verify the construct. So, we fully comply with our Mixed Methods approach.

Confirmability & Statistical conclusion. All items were discussed within the first panel, through different phases, which led to a continuous check. After the whole qualitative research process, factors were evaluated by a large target-panel composed of 124 experts. We computed our results with MS Excel and representative sample sizes to increase statistical power. Moreover, measures and treatment implementation are considered reliable.

It was concluded that the threats would not to be regarded as critical. As we know, there is a constant trade-off between internal and external validity [29], and our sampling strategy took this into account. With Mixed Methods we aim to get useful insight for theory building (external validity and transferability), while we also try to validate it within the same study (internal validity and credibility).

VI. Conclusions

The picture which came out of our research about the Italian IT banking sector is worrisome. A short-term vision drives the strategy about the maintenance and evolution of banking information systems. Our Mixed Methods study used a qualitative Delphi-like methodology to identify 15 concerns which were validated through a survey of a broad target-panel. Concerns were mapped within the ISO 25010 standard. A new *meta quality dimension* emerged from our inductive research approach regarding the Italian IT banking sector. This meta dimension impacts directly both on the architectural drivers and also on software quality, finding a link between ISO 25010 and ISO 42010 standards.

Further research will go on in three main directions. Mixed Methods research is needed to better define meta quality and validate its impact on information systems. Methodological and technical solutions have to be researched to cope with the problems we found. We plan to carry out a similar research at European level, in order to understand how the identified factors impact on other banking information systems.

Acknowledgments

This research has being partially funded by the Consorzio Interuniversitario Nazionale per l'Informatica (CINI) and the Italian National Research Council (CNR-ISTC). The authors thank all the panelists for their time and care in answering to our questions.

References

[1] J. Creswell, *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage, 2013.

[2] D. Shepherd, K. Damevski, and L. Pollock, "How and when to Transfer Software Engineering Research via Extensions," in *Proc. 37th Int. Conf. on Software Engineering*, ser. ICSE, 2015, pp. 239–240.

[3] N. Gorla and S. C. Lin, "Determinants of software quality: A survey of information systems project managers," *Information and Software Technology*, vol. 52, no. 6, pp. 602–610, 2010.

[4] J. Tian, "Quality-evaluation models and measurements," *IEEE Software*, vol. 21, no. 3, pp. 84–91, 2004.

[5] N. Gorla, T. Somers, and B. Wong, "Organizational impact of system quality, information quality, and service quality," *The Journal of Strategic Information Systems*, vol. 19, no. 3, pp. 207–228, 2010.

[6] S. Kan, V. Basili, and L. Shapiro, "Software quality: an overview from the perspective of Total Quality management," *IBM Systems Journal*, vol. 33, no. 1, pp. 4–19, 1994.

[7] B. Slife and R. Williams, *What's Behind the Research?: Discovering Hidden Assumptions in the Behavioral Sciences*. Sage, 1995.

[8] C. Cherryholmes, "Notes on pragmatism and scientific realism," *Educational researcher*, vol. 21, no. 6, pp. 13–17, 1992.

[9] C. Moustakas, *Phenomenological research methods*. Sage, 1994.

[10] P. Ciancarini, D. Russo, A. Sillitti, and G. Succi, "Reverse engineering: a european ipr perspective," in *Proc. 31st Annual ACM Symposium on Applied Computing*, 2016, pp. 1498–1503.

[11] ——, "A guided tour of the legal implications of software cloning," in *Proc. 38th Int. Conf. on Software Engineering*, ser. ICSE, 2016, pp. 563–572.

[12] M. Krafft, K. Stol, and B. Fitzgerald, "How Do Free/Open Source Developers Pick Their Tools?: A Delphi Study of the Debian Project," in *Proc. 38th Int. Conf. on Software Engineering Companion*, ser. ICSE, 2016, pp. 232–241.

[13] T. Chow and D. Cao, "A survey study of critical success factors in agile software projects," *Journal of Systems and Software*, vol. 81, no. 6, pp. 961–971, 2008.

[14] J. Brancheau, B. Janz, and J. Wetherbe, "Key Issues in Information Systems Management: 1994-95 SIM Delphi Results," *MIS Quarterly*, vol. 20, no. 2, pp. 225–242, 1996.

[15] R. Schmidt, K. Lyytinen, M. Keil, and P. Cule, "Identifying software project risks: An international Delphi study," *Journal of Management Information Systems*, vol. 17, no. 4, pp. 5–36, 2001.

[16] E. Doke and N. Swanson, "Decision variables for selecting prototyping in information systems development: A Delphi study of MIS managers," *Information & Management*, vol. 29, no. 4, pp. 173–182, 1995.

[17] U. G. Gupta and R. E. Clarke, "Theory and applications of the Delphi technique: A bibliography (1975–1994)," *Technological forecasting and social change*, vol. 53, no. 2, pp. 185–211, 1996.

[18] R. Schmidt, "Managing Delphi Surveys Using Nonparametric Statistical Techniques," *Decision Sciences*, vol. 28, no. 3, pp. 763–774, 1997.

[19] C. Okoli and S. Pawlowski, "The Delphi method as a research tool: an example, design considerations and applications," *Information & Management*, vol. 42, no. 1, pp. 15–29, 2004.

[20] R. Judd, "Use of Delphi methods in Higher Education," *Technological Forecasting and Social Change*, vol. 4, no. 2, pp. 173–186, 1972.

[21] J. Creswell, V. Clark, and L. Plano, *Designing and conducting Mixed Methods research*. Wiley, 2007.

[22] P. Lavrakas, *Encyclopedia of survey research methods*. Sage, 2008.

[23] W. Brown, R. Malveau, H. McCormick, and T. Mowbray, *AntiPatterns: refactoring software, architectures, and projects in crisis*. John Wiley & Sons, Inc., 1998.

[24] B. Boehm, C. Abts, and S. Chulani, "Software development cost estimation approaches–a survey," *Annals of Software Engineering*, vol. 10, no. 1-4, pp. 177–205, 2000.

[25] J. Zhi, V. Garousi-Yusifoğlu, B. Sun, G. Garousi, S. Shahnewaz, and G. Ruhe, "Cost, benefits and quality of software development documentation: A systematic mapping," *Journal of Systems and Software*, vol. 99, pp. 175–198, 2015.

[26] M. Conway, "How do committees invent," *Datamation*, vol. 14, no. 4, pp. 28–31, 1968.

[27] S. Wagner *et al.*, "The Quamoco product quality modelling and assessment approach," in *Proc. 34th Int. Conf. on Software Engineering*, ser. ICSE, 2012, pp. 1133–1142.

[28] E. Guba, "Criteria for assessing the trustworthiness of naturalistic inquiries," *Educational Communication and Technology Journal*, vol. 29, no. 2, pp. 75–91, 1981.

[29] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Science & Business Media, 2012.