Reasoning and Reuse in Software Architecture Design: Practices in the Argentine Industry

María Celeste Carignano, Silvio Gonnet, Horacio Leone

INGAR / UTN – CONICET, Avellaneda 3657, 3000, Santa Fe, Argentina
{celestec, sgonnet, hleone}@santafe-conicet.gov.ar

Abstract. Over the last years, software architecture design has gained significant importance in both, industrial and research areas due to its relevance in the software system development process. In this context, special attention has been given to the documentation of architects' reasoning during an architectural design, highlighting the advantages and disadvantages of this activity. This work intends to present a view of architects' practices in the Argentine industry regarding reasoning documentation and its subsequent use and access.

Keywords. Survey, Architecture, Software, Reasoning, Reuse, Argentina.

1 Introduction

Over the last years, software architecture design has gained significant importance in both, industrial and academic research areas due to its relevance in the software system development process.

Architectural design is a creative and complex activity performed mainly by software architects. It consists on the implementation of a set of decisions [1] departing from a series of requests and constraints of the system stakeholders. The architect decides which architectural elements will be the most appropriate to build a system that satisfies them. Because of these decisions, it is possible to define the set of structures necessary to reason about the system; these include software elements, the relationships between them and their properties [2].

Nowadays, software architects rely on different methods and methodologies ([3], [1], [4], [5], [6], [7], among others) during the whole process of architecture design or to support a part of it. Moreover, they can apply one or more styles or architectural patterns (such as the ones mentioned in [8]) or use architectural tactics (such as the ones mentioned in [8], [9], [10]) in order to satisfy the expected quality attributes of the system. In other words; nowadays architects can make use of a significant number of options to perform their job.

As architecture design is not standardized, the final architecture strongly depends on who executes it and the methods and tools employed in the design. It is at this point that architects' reasoning during the design process becomes important. The reasoning behind a design decision is the explanation of how and why an artifact, or part of it, has been designed in a given way [11].

In order for system architecture to be truly useful, it has to be understood by those who create, evaluate, use, consult, reuse and service it. This can only be accomplished if the reasoning done during its design is captured and easily retrievable when it is used. Documenting only the generated products is not enough.

Burge et al. [12] identify some key aspects that justify the capture of reasoning in software engineering, which are also valid for architectural design. These are related to:

1. The useful life of software systems: software systems often remain functioning longer than what stakeholders may imagine. For this reason, the software has to be continuously under development in order to continue to be useful. Therefore, it is essential to understand the reasons behind the decisions made before;
2. The iterative nature of software development processes: it requires that decisions made earlier are understood in subsequent steps of the process in order not to infringe or affect them by making new decisions and to evaluate the impacts of updates;
3. The increased participation of stakeholders during the creation of the system, which demands more and better communication to enable the understanding of the decisions made, its scope and impact;
4. Transference of knowledge: significant amounts of experts' knowledge are involved in software system development. This information is lost if it is not documented, especially with the large staff turnovers of the software industry. This problem is known as knowledge vaporization [13], [14];
5. The increase of software systems size and complexity: as systems continuous growth, the reasoning may be used as an *aide-mémoire* for the architects to be reminded why they had made certain decisions. It may also be used to establish the impact that any change in the decisions may cause.

There are many research works ([15], [16], [17], [18], [19], [20], among others) that state and justify the need for the documentation of architects' reasoning and the benefits associated to such activity. Some of the circumstances in which it is possible to notice these benefits are:

- Changes in the initial requirements after the design phase.
- Incorporation of new members into the software construction team.
- Transference of knowledge and/or training of new designers by the architects involved.
- Identification of problems (once the system construction is advanced) that require checking the architecture and analyzing the impact of possible solutions.
- Lack of understanding of the architecture generated.
- Need to generate a new architecture based on earlier experiences so as to avoid repeating previous mistakes and reproduce successes.

Considering what has been theoretically stated, we felt motivated to make a survey directed to architects who work in the software industry in Argentina. Our aim was to learn about their practices as regards reasoning documentation and its subsequent use.

A survey with similar features, although more comprehensive, was carried out by Tang et al. [21] to software architects with more than three years of experience in Asia. Such work enables the comparison between Argentine industry practices and those of a different country. These will be presented throughout this work.

At this point, we will describe how this work is structured. In section 2, we will detail the methodology used to conduct the survey: we will describe the different phases and activities carried out. In section 3, we will present the goals pursued with the survey along with a brief description of the target population. In section 4, we will provide general descriptions about the survey. In section 5, we will present a summary of the characteristics related to the participants. In section 6, we will analyze the data collected and finally, we will discuss the conclusions.

2 Methodology

The survey was conducted in four stages (taking the process described by Pfleeger and Kitchenham in [22], [23], [24], [25], [26] and [27] as reference), which are displayed in a general diagram in Fig. 1.

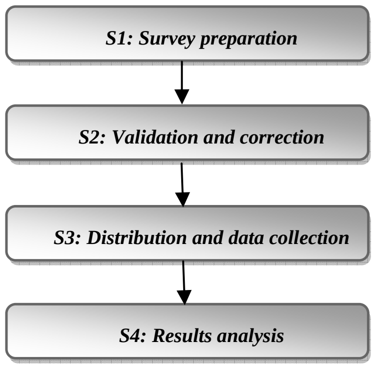
S1: Survey preparation
S2: Validation and correction
S3: Distribution and data collection
S4: Results analysis

Fig. 1. Survey stages

The first stage (*S1* in Fig. 1) was aimed at preparing the survey. Fig. 2 shows the main activities performed to attain that aim. Initially, we defined the goals of the survey (*S1.1* in Fig. 2) by clearly specifying the expected output, and we identified the intended target population (*S1.2* in Fig. 2). We also determined the distribution methodology, and evaluated the multiple tools (free and commercial) to finally select the one that would support the survey performance (*S1.3* in Fig. 2). After that, we created the questionnaire (*S1.4* in Fig. 2) with questions that could collect information that would satisfy the goals previously stated.

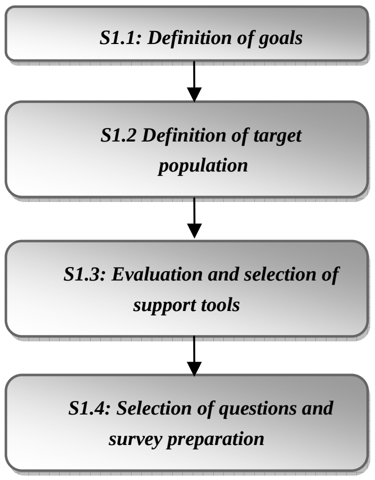
S1.1: Definition of goals
S1.2 Definition of target population
S1.3: Evaluation and selection of support tools
S1.4: Selection of questions and survey preparation

Fig. 2. Activities involved in the survey preparation stage (*S1*)

The second stage (*S2* in Fig. 1) was meant to validate and evaluate the clarity and content of the questions posed in the questionnaire, their distribution and grouping, as well as the ease of use of the selected tool. Fig. 3 describes the activities involved in the implementation of this stage.

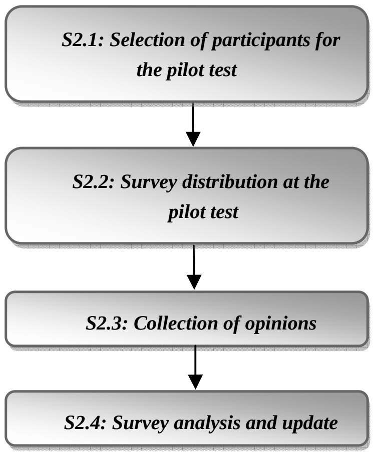
S2.1: Selection of participants for the pilot test
S2.2: Survey distribution at the pilot test
S2.3: Collection of opinions
S2.4: Survey analysis and update

Fig. 3. Activities involved in the validation and correction of the survey stage (S*2*)

A small group of software architects (four members) was selected (*S2.1* in Fig. 3) to participate in the pilot test and one version of the survey was given to them with this purpose (S2.2 in Fig. 3). The comments and suggestions obtained at the pilot test (S2.3 in Fig. 3) were taken into consideration and included in the survey (S2.4 in Fig. 3) to obtain an improved final version.

In the third stage (S3 in Fig. 1), the survey was distributed by contacting several Argentine software architects and inviting them to participate and to act as a link in the dissemination process (S3.1 in Fig. 4). Subsequently, we collected the answers received (S3.2 in Fig. 4).

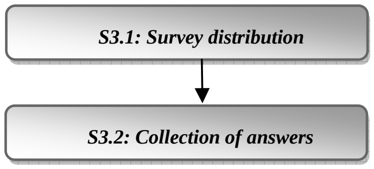

Fig. 4. Activities involved in the distribution and data collection stage of the survey (*S3*).

After the stipulated time, the fourth stage began (*S4* in Fig.1), in which the data collected was analyzed (*S4.1* in Fig. 5) in order to draw conclusions (*S4.2* in Fig. 5), most of which are stated in this work.

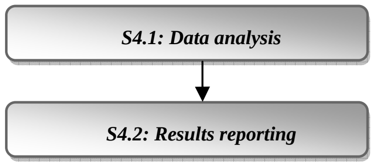

Fig. 5. Activities involved in the survey results analysis stage (*S4*)

3 Objectives of the survey and target population.

The survey was aimed at learning the perceptions and practices of the respondents as regards three important points within the area of software architecture design:

- The documentation of software architects reasoning at decision making during architectural design
- The use of the architectural documented reasoning
- The reuse of architectural design during the definition of new software architecture

The target population was software architects within the Argentine industry.

4 Generalities of the survey

The survey consisted of 28 questions (detailed in Appendix A) grouped in different sections. The main sections were:

i. *General information:* general questions about the respondents which include information about their geographic region, the characteristics of the company where they are currently working (size and certifications), years of experience in the development of software systems in general and in architecture design in particular;
ii. *Reasoning documentation:* questions to learn about the respondents' practices as regards the documentation of their reasoning during architectural design;
iii. *Reuse of architectural designs:* questions to learn about the respondents' practices as regards the reuse of software architecture designs.

The survey was prepared on a tool that would allow access through the web. This form of distribution has several advantages over other, such as those whose support is paper or individual files. Some key advantages are:

- ease of distribution: respondents only need to have a web browser and to know the link to access the web application that contains the survey;
- greater scope of the survey: geographic location is not a constraint to access the survey, therefore, more architects can participate in it;
- greater flexibility for data analysis and elimination of errors that could arise when the answers are processed.

The selected application to support the survey performance was Survey Monkey ([28]), which is a web-based survey tool. The version used was the *Professional* given its support for logic application to the questions (readdressing the respondents to the following question depending on the answer given) and the personalization of the resulting reports, by implementing filters and cross tabulations to the answers.

The invitation to participate in the survey was sent by e-mail to a previously selected group of architects. The e-mail had a link to access the questionnaire and a password. The participants were asked to act as a link by distributing the survey to contacts that fulfilled the previously described characteristics of the target population.

The survey was conducted between the months of February and May of the year 2011.

5 Participants

The survey was targeted at software architecture design workers within software development or consultancy companies in the Argentine industry.

A total of 93 answers were received: 22 of them were incomplete and therefore discarded; 6 of them were excluded from the analysis as they did not satisfy the requisite of having previous experience in software architecture design. Accordingly, 65 answers (69% of the total) were considered in the analysis hereby explained.

The group of participants was heterogeneous regarding their experience in software architecture design and quality standards with which they work at their companies. In Fig. 6 we present the distribution of participants according to the years of experience in architectural design. We can observe that:

- 34 participants (52.3% of the total of respondents) had more than 5 years of experience;
- 13 participants (20% of the respondents) had between 3 and 5 years of experience;
- 13 participants (20% of the respondents) had between 1 and 3 years of experience; and finally,
- 5 participants (7.7% of the respondents) had less than 1 year of experience.

In the software industry, each of these ranks defines the seniority of the participants as experts, senior, semi-senior and junior, respectively.

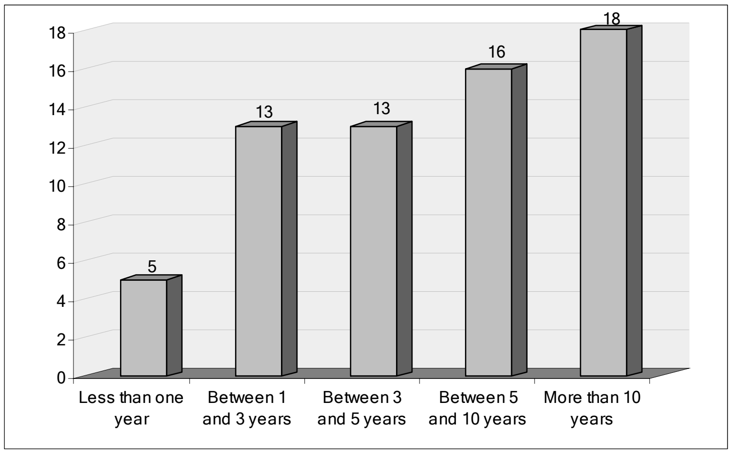

Fig. 6. Number of participants according to their years of experience in software architecture design

As regards the companies in which the participants worked, 72.3% of them (47 people) works in companies that had performed quality certifications such as ISO, CMMI and sometimes both. In Fig. 7 we can observe in greater detail the distribution of the participants according to the type of certifications obtained by the organizations in which they work.

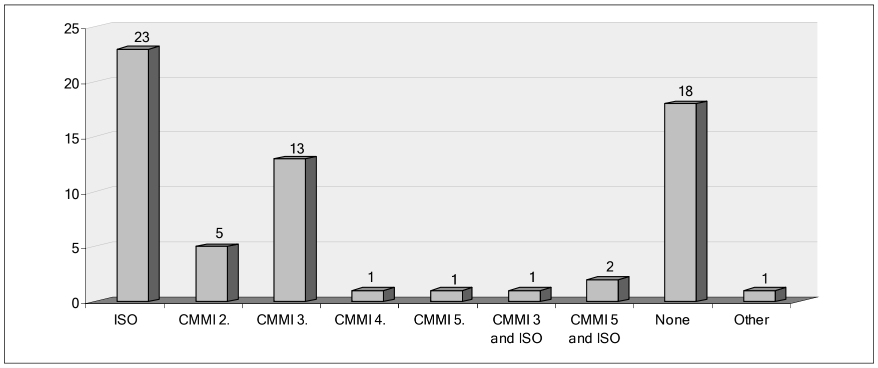

Fig. 7. Number of participants that work at companies that have certified a quality standard, discriminating standard type

6 Data analysis and results

As we have mentioned before, the main goal of this survey was to learn about the practices of the industry related to the capture of architects' reasoning and its subsequent use in both, the same project in which the architecture was built as well as in subsequent ones.

In this section, we will describe the results obtained and analyze them briefly considering the intended goal.

6.1 Reasoning documentation

From the total number of participants, 58.5% of them (38 respondents) admitted documenting their reasoning during software architecture design, whereas the remaining 41.5% stated they did not document such information.

6.1.1 Reasons for not documenting reasoning

In order to better understand why some architects did not document the reasoning behind their decisions, we consult them the reasons behind this omission.

77.8% (21) of the participants that answered they did not document their reasoning said that the main reason for this was lack of time; 7.4% (2) of the participants stated that they do not have the necessary tools to do the job and secure the documentation;

and 14.8% (4) of the participants declared that they do not consider it useful or necessary. Fig. 8 describes this situation.

Up to this point, we may observe that only 4 of the respondents (6.15% of the total of participants) do not consider it useful to document their reasoning. This demonstrates that the remaining 93.85% of the architects reckons the importance of documenting the reasoning behind design decisions made during the definition of architecture, and recognizes the benefits of such a practice.

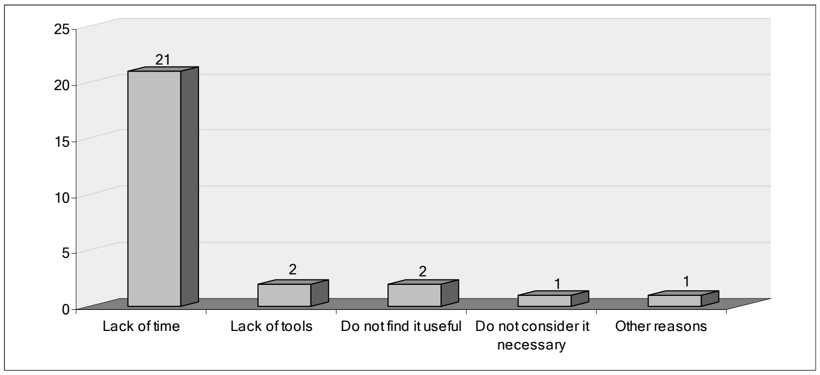

Fig. 8. Number of participants who do not document their reasoning during architectural design, discriminating according to the reasons given

We may find a similarity in this particular point with the survey conducted by Tang and others [21], as it also identifies lack of time as the main reason for not documenting design reasoning. Furthermore, in that survey, similar to the result obtained in this one, there were no participants to consider the reasoning behind design decisions is not important. However, the participants assigned different level of importance to the design rational.

6.1.2 Tools used for reasoning documentation

Another aspect surveyed was related to the tools used to document the reasoning made by the architects. As we can observe in Fig. 9, the main method with which architects document their reasoning is by notes added to the diagrams that describe the architecture, using the same tool they use for designing (50% of the respondents). Documentation in files is another method, both digitally (31.6% of the respondents) as well as manually (10.5% of the respondents). Finally, 7.9% of the respondents employ other means such as Wiki sites.

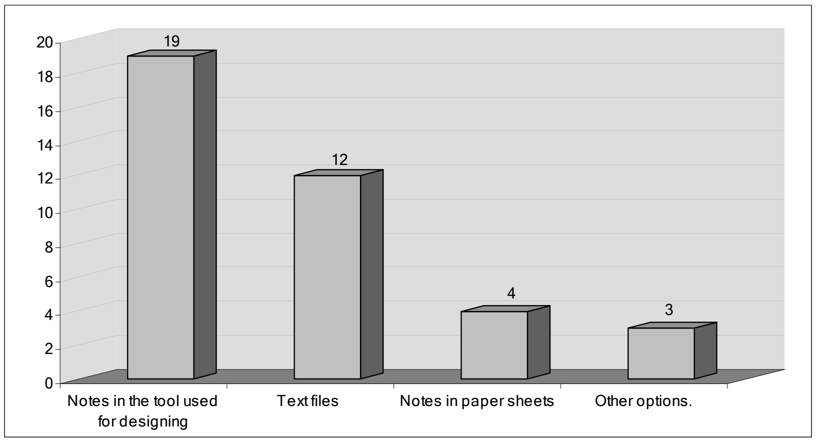

Fig. 9. Main tools used for the documentation of architectural reasoning

In this case, as in the work presented by Tang et al. [21] (in which architects also used the tools previously described for the documentation of their reasoning, among others), we can corroborate that one of the main shortcomings in the area of architectural reasoning documentation is the lack of standards and useful tools in the industry for the documentation of architects' reasoning. Although there are several tools and methods proposed for this purpose, these have not been adopted by architects and, therefore, everybody documents their reasoning according to their needs and experiences.

The fact that the automated tools integrated to the IDE used for designing are not employed helps us understand the result obtained in section 6.1.1: documentation of reasoning by means of free text involves a significant investment of time, both for its capture and its subsequent query, and that is why architects decide not to do it.

6.1.3 Use of the documented reasoning

The capture of the reasoning made by architects is a practice that consuming time and resources. For this reason, it must have several benefits and utilities to justify its realization. With this in mind, the architects were consulted about the use of the documented information.

86.8% (33) of the participants that assured having documented their reasoning during architecture design said that they consult that information later on, whereas 13.2% (5) of the participants stated that they do not access that information again.

The participants who document their reasoning and then consult it do it mainly in the following cases:

- to explain the system architecture to the members of a work team (78.8% of the answers);
- to justify the decisions made (75.8% of the answers);

- to evaluate the architecture and to analyze whether it fulfills new requirements or changes (66.7% of the answers);
- to train other architects (63.6% of the answers);
- to understand somebody else's architecture design (42.4% of the answers).

6.1.4 Analysis and documentation of design alternatives

During the architectural design, the architect must make several decisions. Some of which may require the evaluation of more than one alternative design solution and the selection of one of them. Knowing the reasoning behind of these decisions is important when the architecture is evaluated, analized or modified. Taking this into account, participants were also asked if during architecture design, they evaluated more than one solution alternative. 90.8% (59) of the answers were affirmative; however, only 50.8% (30) of the participants documented the reasoning behind the refusal of the different analyzed solutions.

76.6% (23) of the participants admitted that they later consult the reasoning behind the selection of design alternatives mainly to justify the decisions made (91.3% of the cases).

6.2 Reuse of architectural designs

The architects that participated in the survey were not only asked to give information about the way in which they used the underlying reasoning of the design decisions made, but they were also inquired about the utility they could gain from entire architecture documentation (artifact as well as reasoning) when working on new designs.

From the total number of participants, 98.5% (64 respondents) admitted reusing those architectural solutions employed in previous system architecture that had similar properties to the one they are currently designing. The remaining 1.5% denied performing such an activity as they do not have the necessary information to accomplish it.

65.7% of the respondents (42 participants) attempt to reuse architectural solutions all or at least most of the times they are working on architectural design. Fig. 10 shows the frequency in which the participants attempt to apply reuse mechanisms in software architecture design.

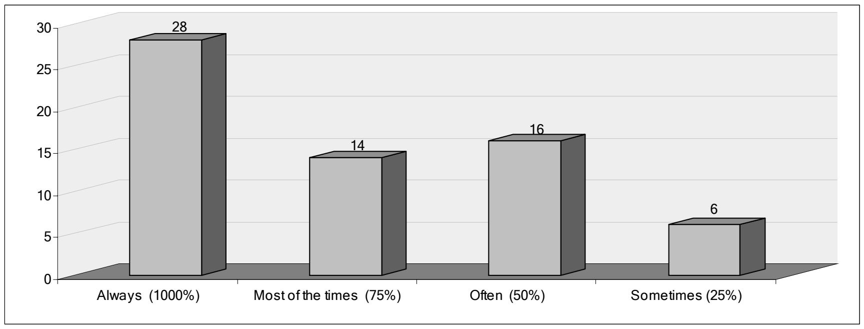

Fig. 10. Frequency of reuse architectural design attempts

In order to determine if the new system properties are similar to those of earlier systems, those who reuse previous architectural solutions consider aspects related to:

- functional requirements (67.2% of the respondents);
- non-functional requirements (59.4% of the respondents);
- design constraints (57.8% of the respondents);
- application domains (42.3% of the respondents);

among other properties. This case is depicted in Fig. 11.

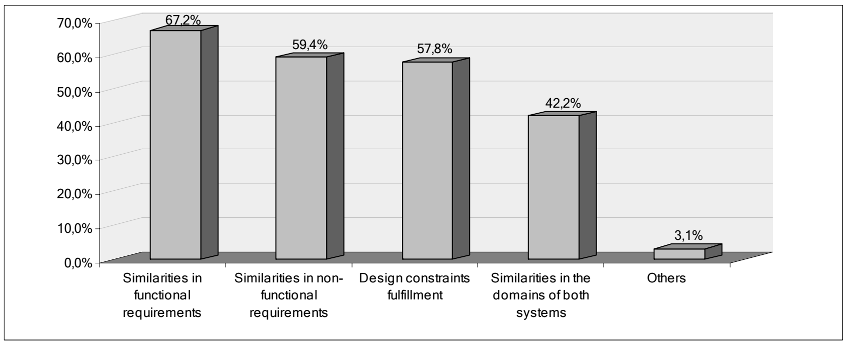

Fig. 11. Criteria considered when reusing architectural designs according to the percentage of participants

In the cases in which the participants access those system architectures that have similar properties to the new system, the information usually reused is:

- Architectural styles implemented (60.9% of the respondents);
- Architectural tactics implemented (50% of the respondents);
- Reasoning to reach the solution (62.5% of the respondents);
- Considerations taken into account (48.4% of the respondents);
- Designed components (48.4% of the respondents).

Fig. 12 portrays this case.

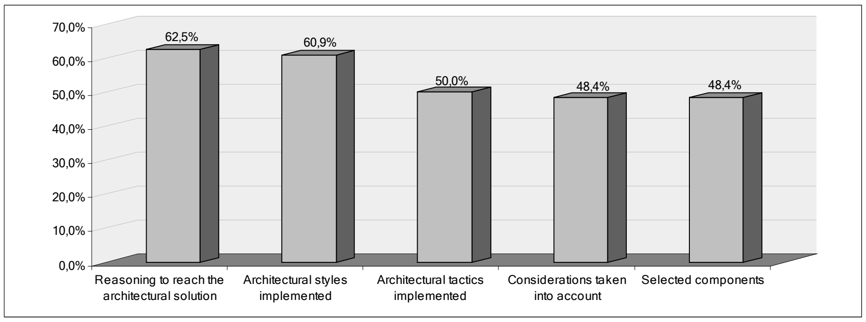

Fig. 12. Information reused when designing new software architecture

The answers obtained demonstrate the importance of reasoning documentation since, at the moment of reusing an architectural solution, not only the components and elements included in the design are considered, but also the decisions made and the reasons behind them.

7 Conclusions

In this work we have presented the results obtained in a survey directed to software architects of the industry in Argentina. They were consulted about their perceptions and practices associated to the documentation of the reasoning underlying design decisions and their subsequent use.

Most of the respondents have shown that they know and attach importance to document reasoning, although the time cost of this practice is an evident limitation for many of them.

As we stressed in previous sections, one of the main problems in this area is related to the lack of standards and tools that enable reasoning documentation.

Consequently, the need for a standard that enable the architect to guide and structure, as much as possible, the reasoning documentation, as well as a tool that ease the adoption of this standard for architects that work in the software industry become apparent. It is important to notice that agility and ease of use are essential requirements in this application.

As a possibility for a future work instance, we propose improving the documentation reasoning process so as to examine/corroborate these points.

Acknowledgements. The authors wish to acknowledge the financial support received from CONICET, Universidad Tecnológica Nacional and Agencia Nacional de Promoción Científica y Tecnológica (PAE-PICT 2315, IP-PRH 2007).

Bibliografía.

1. Jansen, A., Bosch, J. Software architecture as a set of architectural design decisions. In: WICSA '05: Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture. Washington, DC, USA: IEEE Computer Society. ISBN 0-7695-2548-2; 2005:109-120. doi: http://dx.doi.org/10.1109/WICSA.2005.61.
2. Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R., Stafford, J. Documenting Software Architectures: Views and Beyond (2nd Edition). Addison-Wesley Professional; 2010. ISBN 978-0321552686.
3. Wojcik, R.; Bachmann, F.; Bass, L.; Clements, P.; Merson, P.; Nord, R.; Wood, B. Attribute-Driven Design (ADD), Version 2.0, (CMU/SEI-2006-TR-023). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University (2006).
4. Clements, P., Kazman, R., Klein, M.: Evaluating Software Architecture, Addison-Wesley, Boston (2002).
5. Kruchten, P: The 4 + 1 view Model of Architecure. IEEE Software 12 (6), 45-50 (1995)
6. Hofmeister, C., Kruchten, P., Nord, R., Obbink, H., Ran, A., America, P.: Generalizing a model of software architecture design from five industrial approaches. In: Proceedings of 5th Working IEEE/IFIP Conference on Software Architecture (WICSA 5), Pittsburgh, PA, IEEE Computer Society, 77-86 (2005).
7. Kruchten, P.: The Rational Unified Process: An Introduction, third ed. Addison-Wesley, Boston (2003).
8. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, Addison-Wesley Professional; 2 edition (2003).
9. Bachmann, F.; Bass, L.; and Nord, R. Modifiability Tactics, (CMU/SEI-2007-TR-002). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, (2007).
10. Scott, James, and Rick Kazman. Realizing and Refining Architectural Tactics: Availability (CMU/SEI-2009-TR-006). Pittsburgh, PA: Software Engineering Insitute, Carnegie Mellon University (2009).
11. T.R. Gruber and D.M. Russell. Design knowledge and design rationale: A framework for representing, capture, and use. Technical report, Knowledge Systems Laboratory, Standford University, California, USA, 1991.
12. J. Burge, J. Carroll, R. McCall, and I. Mistrík. Rationale-Based Software Engineering. Springer-Verlag, 2008.
13. Jansen, A, Bosch, I. . Software architecture as a set of architectural design decisions. In: WICSA '05: Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture., pages 109-120, 2005.
14. Ven, J., Jansen, A., Nijhuis J., Bosch, J.. Design Decisions: The Bridge between Rationale and Architecture. pages 329{348, 2006.
15. van der Ven, J. S., Jansen, A., Nijhuis, J., Bosch, J.: Design Decisions: The Bridge between Rationale and Architecture, chapter 16, Rationale Management in Software Engineering, A. Dutoit, R. McCall, I. Mistrkik, and B. Paech (Eds.), Springer-Verlag, 329-346 (2006).
16. Babar, M., Gorton, I., Kitchenham, B.: A Framework for Supporting Architecture Knowledge and Rationale Management, chapter 11, Rationale Management in Software Engineering, A. Dutoit, R. McCall, I. Mistrkik, and B. Paech (Eds.), Springer-Verlag, 237-254 (2006).
17. Babar, M. A., Gorton, I.: A Tool for Managing Software Architecture Knowledg. Second Workshop on SHAring and Reusing architectural, Knowledge Architecture, Rationale, and Design Intent (SHARK-ADI'07) (2007)

18. Bass, L., Clements, P., Nord, R. L., Stafford, J.: Capturing and Using Rationale for a Software Architecture, chapter 12, Rationale Management in Software Engineering, , A. Dutoit, R. McCall, I. Mistrkik, and B. Paech (Eds), Springer-Verlag, 255-272 (2006).
19. Tang, A., Jin, Y., Han, J.: A rationale-based architecture model for design traceability and reasoning. The Journal of Systems and Software 80, Elseiver, 918-934 (2007)
20. Jasen, A., van der Ven, J., Avgeriou, P., Hammer, D.: Tool support for Architectural Decisions, Proceedings of the Sixth Working IEEE/IFIP Conference on Software Architecture (WICSA'07), 44-53, (2007).
21. Tang, A., Muhammad, A. B., Gorton, I., Jun, H. A survey of architecture design rationale. J. Syst. Softw. 79, 12, 1792-1804 (2006).
22. Pfleeger, S.L., Kitchenham, B. A.: Principles of Survey Research – Part 1: Turning Lemons into Lemonade, Software Engineering Notes vol 26 no 6, 16-18 (2001).
23. Kitchenham, B. A., Pfleeger, S.L.: Principles of Survey Research – Part 2: Designing a Survey, Software Engineering Notes vol 27 no 1, 18-20 (2002).
24. Kitchenham, B. A., Pfleeger, S.L.: Principles of Survey Research – Part 3: Constructing a Survey Instrument, Software Engineering Notes vol 27 no 2, 20-24 (2002).
25. Kitchenham, B. A., Pfleeger, S.L.: Principles of Survey Research – Part 4: Questionnaire Evaluation, Software Engineering Notes vol 27 no 3, 20-23 (2002).
26. Kitchenham, B. A., Pfleeger, S.L.: Principles of Survey Research – Part 5: Populations and Samples, Software Engineering Notes vol 27 no 5, 17-20 (2002).
27. Kitchenham, B. A., Pfleeger, S.L.: Principles of Survey Research – Part 6: Data Analysis, Software Engineering Notes vol 28 no 2, 24-27 (2003).
28. Survey Monkey, disponible en: http://es.surveymonkey.com (2012).

Appendix A

Below are the questions that were part of the questionnaire.
“The company” is the company where you are currently employed.

1. How many employees does the company have?
- Lest than 10 employees.
- Between 11 and 20 employees.
- Between 31 and 50 employees.
- Between 51 and 100 employees.
- Between 101 and 200 employees.
- More than 200 employees.

2. Does the company have any quality certification?
- Yes, ISO 9001.
- Yes, CMMI Level 2.
- Yes, CMMI Level 3.
- Yes, CMMI Level 4.
- Yes, CMMI Level 5.
- No.
- Other.

3. In which state is located the company (or office)?

4. How long have you been developing software?
- I have never been involved in the development of software.
- Less than a year.

- Between 1 and 3 years.
- Between 3 and 5 years.
- Between 5 and 10 years.
- More than 10 years.

5. How long have you been designing software architectures?
- I have never been involved in the design of software architecture.
- Less than a year.
- Between 1 and 3 years.
- Between 3 and 5 years.
- Between 5 and 10 years.
- More than 10 years.

6. How many software architecture designs have you been involved in?
- None.
- Less than 5.
- Between 5 and 10.
- Between 10 and 20.
- More than 20.

7. Have you got any training in architectural design? (multiple selection allowed)
- Yes, self-training.
- Yes, I have taken internal courses provided by the organization in which I work for.
- Yes, I have taken courses provided by experts in the subject.
- No.
- Others

8. Regarding to the architecture design process, which is the type of requirement that impact the most over the architecture?
- Functional requirements.
- Quality requirements.
- Design restrictions.

9. When you are working with the non-functional requirements, Do you prepare a priority ranking to manage these requirements during the architectural design?
- Yes, the established by the client.
- Yes, the established by the client but taking in count my experience.
- Yes, the one I consider most suitable.
- Yes, I consider first that ones that I know the most and have more knowledge resolving its.
- I do not make any priorities, I start working at random.
- Other.

10. Which are the non-functional requirements that are more important when a particular architecture is designed? (multiple selection allowed)
- Performance.
- Modificability.
- Portability.
- Security.
- Reliability.
- Availability.
- Easy testing.
- Usability.
- Others.

11. Which are the most common design constraint ? (multiple selection allowed)
- Programing language restrictions.
- Data Base restrictions.
- Restrictions in the use of particular technologies.
- Communication restrictions.
- Components restrictions.
- Others.

12. Are the design constraints imposed by the user?

- Yes, all of them.
- Yes, but some of them are suggested by me when the requirements are evaluated, before I start with the architecture.
- No.
- Other.

13. Which languages do you use to design the architectures? (multiple selection allowed)
- UML.
- OCL.
- Textual description.
- Graphic notation different from UML.
- Especific ADL.
- If you use any ADL, Could you indicate which one?

14. What tool do you use to design the architecture?
- Enterprise Architect.
- Rational.
- Visual Studio.
- Text editor.
- Other.

15. Do you document the reasoning doing during the software architecture design?
- Yes.
- No.

16. How do you document reasoning doing during the software architecture design?
- Text files.
- Notes in the tool used for the design.
- Notes in sheets.
- Others.

17. Do you use the documentation of the reasoning doing during the software architecture design in another moment?
- Yes.
- No.

18. When do you use the documentation of the reasoning doing during the software architecture design? (multiple selection allowed)
- When I have to explain the final architecture to the rest of the project team.
- When I have to train others architects.
- When I have to justify my choices.
- When I have to analyze if the architecture satisfied new requirements or changes needed by the user.
- When I have the role of architect in a project if which the architecture has been designed by another person.
- Other.

19. Why do you not document the reasoning doing during the software architecture design?
- I consider that is not necessary.
- I consider is not useful.
- Although I consider is necessary, I do not have the tools.
- Although I consider is necessary, I do not have the time.
- Other.

20. When you are designing the architecture, Do you evaluate more than one possible solution (alternatives)?
- Yes.
- No.

21. Do you document the reasoning doing when you discard or reject proposed and analyzed solution alternatives?
- Yes.
- No.

22. Do you use the documentation of the reasoning doing when you discard or reject proposed and analyzed solution alternatives in other moment?
- Yes.

- No.

23. When do you consult the documentation of the reasoning doing when you discard or reject proposed and analyzed solution alternatives? (multiple selection allowed)
- When I have to explain the final architecture to the rest of the project team.
- When I have to train others architects.
- When I have to justify my choices.
- When I have to analyze if the designed architecture allow new requirements or changes needed by the user.
- When I have the role of architect in a project in which the architecture has been designed by another person.
- Other.

24. When you are designing a new system architecture, do you use similar architectural solutions used in systems which have similarities which the new one?
- Yes.
- No.

25. How do you determine that the characteristics of the new systems are similar to previous systems? (multiple selection allowed)
- I analyze if the functional requirements are similar.
- I analyze if the systems' domains are similar.
- I analyze if the non-functional requirements are similar.
- I analyze the imposed design restrictions and search for applied solutions in the previous architecture.
- Other.

26. If you find systems with similar characteristics, what architectural information do you find useful? (multiple selection allowed)
- Architectural styles applied.
- Architectural tactics applied.
- The reasoning applied to obtain the architectural solution.
- Considerations taking into account.
- Selected components.
- Other.

27. How frequenly do you analyze if there are previous architectural solutions that can be used to design a new architecture?
- For all the systems in which an architecture has to be designed (100%).
- For most of the systems in which an architecture has to be designed (75%).
- For some of the systems in which an architecture has to be designed (50%).
- For a few of the systems in which an architecture has to be designed (25%).

28. Why do not you analyze if there are previous architectures that can be useful to design a new architecture? (multiple selection allowed)
- I have not got the information.
- I have no interest in it.
- I have never found it useful.
- Others.