Software Requirement Patterns

Xavier Franch
Universitat Politècnica de Catalunya (UPC)
Barcelona, Spain
franch@essi.upc.edu

***Abstract*—Software requirements reuse becomes a fundamental activity for those IT organizations that conduct requirements engineering processes in similar settings. One strategy to implement this reuse is by exploiting a catalogue of software requirement patterns (SRPs). In this tutorial, we provide an introduction to the concept of SRP, summarise several existing approaches, and reflect on the consequences on several requirements engineering processes and activities. We take one of these approaches, the PABRE framework, as exemplar for the tutorial and analyse in more depth the catalogue of SRP that is proposed. We apply the concepts given on a practical exercise.**

***Index Terms*— Software Requirements Patterns, Requirements Engineering, Patterns, Software Reuse.**

I. Motivation

Recent reports show how still today a significant percentage of software projects are out of budget, suffer delays or simply have to be cancelled. One of the most recognized causes for this scenario is the failure in producing a good set of software requirements [1]. Several studies have explored this link throughout the last decades. As far as in 1981, Boehm already mentioned that approximately 60% of all errors in system development projects originate during the phase of requirements engineering [2]. Also the high cost of fixing requirement errors was stated in that work. Even if the requirements engineering field has substantially improved since then, still it causes problems in industrial projects. A recent Gartner report [3] states that requirements defects are the third source of product defects (following coding and design), but are the first source of delivered defects (in particular for service projects). Other recent studies [4] report that the definition of requirements remains a grand challenge for IT professionals. Methods for improving the quality of software requirement specifications are therefore still needed.

Pattern-based requirements engineering is one of such methods. The definition and use of a software requirements pattern catalogue can support the elicitation, validation, documentation and management of requirements. By designing an appropriate catalogue, an IT organization will have a starting point for the requirements engineering activity reducing the associated costs and producing better requirements.

In this work, a tutorial on software requirement patterns is outlined. It was taught at the ICSE 2013 conference. The tutorial introduces the foundations of software requirement patterns (SRPs): concept, metamodel, semantics, classification schemas and processes. Concrete objectives of such a tutorial are:

- Understanding the concept of SRP as a reuse unit.
- Mastering the processes around SRPs: construction and evolution of an SRP catalogue, and its application on a particular project.
- Realizing the difference among functional (heavily domain-dependent) and non-functional (quite domain-independent) requirements concerning SRP definition.
- Having a look to a subset of SRPs as a way to better understand the practical implications of the concept.
- Acquiring proficiency enough to apply SRPs in a particular organizational setting.
- Uncovering the insights and lessons learned of a large-scale case study of construction of an SRP catalogue.

The tutorial is addressed to researchers, practitioners and educators in software engineering, especially requirements engineers. For researchers, an updated state of the art is exposed, whilst the presentation relies on scientific grounds. For practitioners, processes and templates are shown and a successful case study of pattern-based requirements engineering is analysed in detail. For educators, the tutorial provides the basis for developing course material.

II. Background

Reuse is a fundamental dimension in all software engineering disciplines, and requirements engineering is not an exception [5]. A good approach to software requirements reuse may help requirement engineers to efficiently elicit, validate and document software requirements and as a consequence, obtain software requirement documents of better quality both in contents and syntax [6].

From the several possible approaches to reuse, we are interested in the adoption of patterns [7]. Software engineering practitioners have adopted the notion of pattern in several contexts, remarkably related with software design (e.g., design patterns and software architectural patterns), but also in other software development phases, both earlier and later. In this tutorial, we provide insights in the use of patterns in the requirements engineering phase, namely Software Requirement Patterns (SRP).

An SRP is a guide for writing a particular type of requirement [8]. There are several perspectives that are fundamental in a proposal of SRPs which will be discussed in

the tutorial: theoretical (metamodels, ontologies), methodological (processes around) and organizational (impact, cost analysis).

The importance of requirements engineering (RE) in software engineering has been documented from long ago. Clearly, both characteristics together point out about the convenience of methods and strategies to improve RE practices. SRP fall into this category.

The most influential book on the topic is John Withall's book [8]. Classical Robertson and Robertson's book [6] also includes some sections of direct interest. A long tradition of papers on software requirement patterns exists in the IEEE RE[1] and REFSQ[2] conferences. Recently, the Requirement Patterns workshop[3] (RePa), held at IEEE RE, has been launched, with a good number of contributions and attendees in its two editions (third one is currently on the way).

The tutorial is based on the author's experience, as scientific leader of the GESSI research group at the UPC[4], on developing a requirement patterns program together with the CITI department at the Centre de Recherche Publique Henri Tudor (CRPHT) at Luxembourg. The result is the PABRE framework, explained in detail in a dedicated webpage[5]. It includes: a general explanation; the current catalogue of patterns; presentation of the two existing subsystems (pattern management and pattern application); envisaged collaboration schemas for the use of the catalogue, the method and the tool; list of publications.

III. CONTENTS

A tutorial like this, highly methodological and aiming at being practical, needs to be composed of a theoretical part and a practical part. The theoretical part shall contain also short 2-3 minutes exercises to make the attendees aware of their progress and also to make the tutorial more entertaining. In this outline, emphasis is put on the theoretical part. The topics proposed are: motivation, concept of SRP, structure of SRPs, classification schemas for SRP catalogues, analysis of an existing framework of SRPs and presentation of processes around SRP catalogues. We present some details of these topics below (except for motivation already introduced in Section II).

A. Concept of SRP

Adapting Alexander's seminal definition [7], each SRP describes a requirements' need (the problem) which occurs over and over again in a given context, and then describes a set of interrelated high-level, customizable requirements (the solution) to that need, in such a way that it can be used a million times over, without ever doing it the same way twice.

In a literature review, we found several approaches to SRP that may be classified along different dimensions: (1) the activity where they are used (e.g., requirement elicitation and documentation [10], knowledge management [11]); (2) they may be domain-dependent (e.g., [12] for embedded systems) or generic (e.g., Withall's catalogue [8]); (3) they may be expressed in natural language [8, 9] or some other formal [12] or at least rigorous [11, 13] notation; (4) they may be thought for a particular type of project (e.g., call for tender processes [9]) or not; (5) their structure may be informally defined [8] or compliant to some grammar [14] or metamodel [10].

B. Structure of an SRP

The classical approach by Whitall proposes the following structure [8]:

- Basic details: manifestation, domain, related patterns (if any), anticipated frequency of use, classification, author.
- Applicability. Context of application.
- Discussion. Implications of the type of requirement targeted by the SRP.
- Content. Core of the pattern, explains what is necessary to state for this type of requirement.
- Template. Starting point for writing requirements of this type.
- Examples.
- Extra requirements. Requirements that may be implied by the main one.
- Considerations for development. Hints for software engineers and designers on how to implement a requirement of this type.
- Considerations for testing. Similar but for testing.

This structure needs to be considered a general indication and by no means the only possible approach to implement an SRP-based approach.

C. Classification Schemas for SRP Catalogues

A fundamental issue when considering SRPs as part of a catalogue is the need of classifying them over some criteria for supporting their search. In fact, it is important to observe that different contexts (organizations, projects, standards, etc.) may, and usually do, define or require different classification schemas. History shows that trying to impose a particular classification schema does not work, therefore we adopt the position of considering the catalogue a flat structure such that different classification schemas may be defined on top. A classical quality hierarchy as e.g. ISO/IEC 25010 [15] may be used as classification schema.

D. Analysis of the PABRE Framework

The PABRE framework has been defined as result of a cooperation among the GESSI research group at UPC and the SRI department at CRPHT. In relation to this topic, CITI's mission consists on helping SME with no background in requirements engineering to handle requirements analysis activities and to design SRS in order to conduct call-for-tender processes for selecting Off-The-Shelf (OTS) solutions [16]. More than 40 projects ran successfully following the CITI methodology, but the only technique of reuse they applied was starting a new project by editing the most similar requirement book. These techniques demonstrated their weaknesses

[1] http://requirements-engineering.org/
[2] http://refsq.org/2013/past-conferences/
[3] http://www.utdallas.edu/~supakkul/repa12/
[4] http://www.essi.upc.edu/~gessi/
[5] http://www.upc.edu/gessi/PABRE/index.html/

especially in relation to mobility of IT experts and consultants. It became necessary to provide better means to capitalize requirements in a high-level manner by creating reusable artifacts like patterns, supporting consultants' need of creating new SRS. This was the seed of the PABRE framework [17].

The PABRE framework consists of:

- The customization of the concept and structure of SRP to the business case in which CITI is implied.
- A metamodel that defines rigorously the concept and structure and facilitates tool support [10].
- A comprehensive catalogue that includes a set of 29 SRP patterns for non-functional requirements [9], 37 for non-technical requirements [18] and a set of SRP for functional requirements in the domain of context management systems [19].
- A method for the use of the SRP catalogue in requirements elicitation and documentation [9].
- A software system for the catalogue use, management and evolution [20]. This tool is public and several usage schemas have been designed [21].

E. Processes around SRP Catalogues

An SRP catalogue is used during the requirement elicitation phase of IT systems and services projects. During this use, requirement engineers select SRP from the catalogue that apply to the particular project and convert them into the real requirements that finally configure the software requirement specifications. In a nutshell, working with SRPs converts requirement elicitation into a process of search in, and pick-up from, the SRP catalogue, and then make the necessary adaptations to the project context.

IV. CONCLUSIONS

This tutorial provides the basis for understanding the concept of software requirement patterns and the implications in requirements engineering processes. Said that, it is clear that every IT organization will have its own peculiarities which will lead to different instantiations of the generic concept and processes. Small organizations may opt by more lightweight solutions, whilst big companies that handle dozens of projects a year with lots of similarities may benefit from a more rigorous and staged approach.

ACKNOWLEDGMENT

This work is partially supported by the Spanish research project TIN2010- 19130-C02-01. Also the author wants to thank the members of the GESSI and SRI teams involved in this project.

REFERENCES

[1] H. Hofmann, F. Lehner. "Requirements Engineering as a Success Factor in Software Projects". *IEEE Software*, 18(4), 2001.

[2] B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

[3] Gartner Group. *Hype Cycle for Application Development: Requirements Elicitation and Simulation*, July 2011.

[4] SwissQ Consulting. *SwissQ Requirements Trends & Benchmarks Switzerland*, 2012.

[5] W. Lam, J. A. McDermid, A. J. Vickers. "Ten Steps towards Systematic Requirements Reuse". REJ 2(2), 1997.

[6] S. Roberson, J. Robertson. *Mastering the Requirements Process* (2nd ed.). Addison-Wesley, 2006.

[7] C. Alexander. *The Timeless Way of Building*. Oxford Books, 1979.

[8] S. Withall. *Software Requirements Patterns*. Microsoft Press, 2007.

[9] X. Franch, C. Palomares, C. Quer, S. Renault. F. de Lazzer. "A Metamodel for Software Requirement Patterns". REFSQ 2010.

[10] S. Renault, O. Méndez, X. Franch, C. Quer. "A Pattern-based Method for building Requirements Documents in Call-for-tender Processes". IJCSA 6(5), 2009.

[11] J. Yang, L. Liu. "Modelling Requirements Patterns with a Goal and PF Integrated Analysis Approach". COMPSAC 2008.

[12] S. Konrad, B.H.C. Cheng. "Requirements Patterns for Embedded Systems". RE 2002.

[13] S. Supakkul, T. Hill, L. Chung, T.T. Tun, J. Leite. A"n NFR Pattern Approach to Dealing with NFRs". RE 2010.

[14] K. Watahiki, M. Saeki. "Scenario Patterns Based on Case Grammar Approach". ISRE 2001.

[15] The ISO Organization. *ISO/IEC 25010: Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models*, 2011.

[16] M. Krystkowiak, B. Bucciarelli. "COTS Selection for SMEs: a Report on a Case Study and on a Supporting Tool". RECOTS 2003.

[17] X. Franch, C. Quer, S. Renault, C. Guerlain, C. Palomares. "Constructing and Using Software Requirement Patterns". In W. Maalej, A.K. Thurimella (eds.), *Managing Requirements Knowledge*, Springer, 2013.

[18] C. Palomares, C. Quer, X. Franch, C. Guerlain, S. Renault, "A Catalogue of Non-Technical Requirement Patterns". RePa 2012.

[19] C. Palomares, C. Quer, X. Franch, S. Renault, C. Guerlain. "A Catalogue of Functional Software Requirement Patterns for the Domain of Content Management Systems". SAC 2013.

[20] C. Palomares, C.Quer, X.Franch, "PABRE-Man: Management of a Requirement Patterns Catalogue". RE 2011.

[21] X. Franch, C. Guerlain, C. Palomares, C. Quer, S. Renault. "Interested in Improving Your Requirements Engineering Process? Try Requirement Patterns!". Empirical Fair Track at REFSQ 2011.