A Potted History of a Requirements Management Training Course

Olivier Hayard
Itecor
Paul Cérésole 24, cp 568
1800 Vevey 1, Switzerland
o.hayard@itecor.com

Abstract

I describe more than fifteen years of experience teaching Requirements Management to practitioners in industry, to software engineering graduate students in academia and to future practitioners as part of Certificate of Advanced Studies in Business Analysis program. I explain the various contexts, constraints, some of the difficulties I experienced with this course and the lessons learned.

1 Introduction

I was involved in the practice of managing requirements for software systems since 1992. I realized only in 2008, when I began to collaborate with researchers at Ecole Polytechnique Fédérale de Lausanne (EPFL), that Requirements Engineering was a well-established academic research discipline. The corresponding practice in industry has several names, including Requirements Management and Business Analysis. In this paper I describe an industry Requirements Management course that I developed since 1999. It was first designed for practitioners in one if the major European automobile manufacturers. It was then adapted to a mixed practice/academic setting format. I discuss how the practice of Requirements Management has evolved during these years and the corresponding evolution of the course. Over this timespan, we have moved from software engineering and waterfall methods to service management and agile methods.

2 The Context

2.1 General

Between 1992 and 1996 object oriented software engineering was a new paradigm in business applications. By new practice I mean recently widely used. We are in an era where most business applications are still written in COBOL and running on mainframes. The emerging trends are client-server architectures and object oriented programming. Large organizations are ready to experiment with these new trends. The limits of the prevailing waterfall model were already becoming visible at the time. New software project development models, such as the spiral model [1] and Rapid Application Development (RAD) [2] were being proposed. The real revolution in software engineering came when we began to implement the "Use Case" technique that became popular with Jacobson et al.'s book [3].

2.2 Personal

I graduated as a software engineer and joined Itecor, a newly founded consultancy firm, in 1993[1]. Itecor's mission was to help companies reengineer their information systems. As Itecor was founded by former Ernst & Young mission directors, I was primarily working on projects using E&Y Navigator Systems Series (E&Y/NSS, see [4]):

Navigator offers a traditional waterfall model for the application development process. When I join Itecor in 1993, I begin using Navigator with RAD, Navigator having added a RAD model for IT custom development projects. As of 1995 we come into contact with Rational Software. Following his company's (Objectory AB) acquisition by Rational Software in 1995, Ivar Jacobson is touring Europe, presenting the use case approach. In March 1996 Itecor participates in a workshop where Ivar Jacobson presents the use case approach. Itecor then decides to partner with Rational Software. We organize a joint event with Rational Software, for Itecor clients, where Jacobson presents a light version of his use case approach, and where Itecor shows a concrete implementation of the use case approach for one of our customers. From this moment on, use cases become the only concept employed from requirements to deployment, in what is known as the Use Case Driven approach. In particular, the

[1] Itecor was called Information Technology Solutions S.A. (abbreviated as IT Solutions) back then

totality of requirements analysis is done through use cases. All the elaborate concepts of Navigator (spanning 106 volumes, probably about 10'000 pages) are quickly forgotten.

At Itecor we then use Rational Rose (the tool) with Objectory 4.0 (the method) for most of our missions. In 1997 Rational Software acquires Requisite Inc. and SQA Inc. In 1998 Rational ships the Rational Unified Process (RUP) version 5.0, which integrates business use cases, requirements and the FURPS software quality attributes model. We now have to manage both traditional use cases, the new concept of business use case and the concept of requirements[2]. As a result, the simplicity of the use case driven approach is superseded by a framework with increased complexity. Our clients expect us to make sense of this complexity by digesting these new tools and methods and helping them to redefine new processes that are compliant with these de facto standards proposed by Rational Software.

In 1999 I am working as a consultant for the methodology team of the IT department of one of the major European automobile manufacturers. This client has a business application development method based on Navigator. The DNA of this client being geared toward automobile manufacturing, this method is itself engineered with industrial grade quality. We need to propose a RUP-based method that is of equal quality to their Navigator-based method, meaning that it can be industrialized just like car manufacturing has been industrialized. It must be repeatable, optimized, documented, etc., i.e., CMM level 5.

One of the members of the methodology team had a background in the aeronautics industry. While brainstorming on the problem mentioned above, we realized that the requirements management process of the construction of an aircraft is similar to the software requirements process. We decide to base the future requirements management process for this automobile manufacturer on this aeronautics industry requirement process. We build a requirements management course, directly derived from this industry-grade method. Beyond the dream, prevalent at the time, of software engineering as an industrial practice, we were astonished to discover how much we could reuse industry-borne practices in requirements management. The complexity of requirements management even looked quite benign compared to the complexity of the construction of a passenger aircraft. Only retrospectively, about 10 years ago, did I discover that this industrialized view of requirements management existed in the form of an industry standard, namely IEEE 1233.

3 The Course

The course was given from 1999 to today in different contexts. In the list below, I summarize the periods during which the course was given and the main aspects of the course. The list shows how the course evolved over the years.

The 1999 – 2000 Version

This course was given at a client site, an automobile manufacturer, to about 10 people in 3 groups. The main aspects of the course were:

- Proof that the envisioned system is the optimal solution for the expressed requirements.
- The requirements engineering process that is taught is implicitly in-line with IEEE 1233. In addition it includes the allocation of requirements to system components.
- The process is illustrated the Automatic Teller Machine (ATM) example, ubiquitous at the time in software engineering.
- Solution validation is done through a process called Inspection Analysis Demonstration Test (IADT), a process prevalent in the aeronautics industry.

The 2001 Version

Given inside Itecor to a team of 10 consultants with no changes from the previous version.

The 2003 – 2005 Version

Given at the HEIG-VD University of Applied Sciences of the Canton of Vaud, Switzerland, to undergraduate engineers. The main changes were:

- Integration of the FURPS+ requirements classification model, replacing the initial proprietary model that categorized requirements into functional, non-functional and constraints.
- Since this is a full academic, graded course, I created an additional example for the examination on top of the ATM. The example was called the PizzaMat, an automated Pizza distribution machine. This became the standard running exercise threading through the whole course.

[2] We also needed to integrate the other RUP 5.0 supporting workflows: configuration and change management

The 2006 – Today Version

Given to various customer BA teams, this course introduced the following parts:

- Requirements management in the context of Quality Assurance, as per ISO 9000. FURPS+ is now presented as just one possible quality model. I now present other alternatives to FURPS+, e.g. ISO 25000, which superseded ISO 9126.
- The system seen as a service, as per ITIL, adding aspects such as service level requirements, change management, release management, configuration management. We are now specifying the requirements of the organization around the system so that the system can be operated to deliver a continuous service. Requirements are now allocated not only to system components, but also to parts of the organization. Release management of components and corresponding requirements allocation is now explicit.
- I added a disclaimer about seeing this requirements engineering process as the silver bullet that insures that we build the right solution. I know insist more on the problems of eliciting requirements, identifying stakeholders and more generally making sure that the requirements for which the system is designed are the right ones. This also includes aspects of change management of the requirements themselves. This PizzaMat exercise is now used, not to specify a system, but to uncover all the implicit requirements that can hide in a typical requirements document. Including the service and organizational components e.g., payment service provider, electricity provider, pizza provider, maintenance, cleaning, food safety.

The 2014 – 2015 Certificate of Advanced Studies in Business Analysis Variant

In 2014 I was asked to give a requirements management module at the University of Applied Sciences of the Canton of Geneva. The module was part of a Certificate of Advanced Studies in Business Analysis. I therefore made two separate variants of the course. One that I maintain as the standard requirements management course and a second, more specialized in business analysis and agile methods. The latter includes the following changes:

- I added several standard definitions sourced from IEEE 610.12-1990, ISO 8402, IIBA BABOK, IREB CPRE Glossary and PMI Business Analysis for Practitioners guide.
- I added the requirements scope in a business context from ISO IEC IEEE 29148-2011 that introduces the hierarchy of requirements from business to applications. The requirements engineering process is applicable to each level the hierarchy just like a succession of Russian dolls.
- I added explicit references to IEEE Std 830-1998 and IEEE 1233-1998 regarding the well-formedness of Requirements.
- I added explicit links with agile methods such as SCRUM, in particular the integration of requirements management and management of the product backlog.

4 Lessons Learned

The requirements engineering process, which is at the heart of the course, did not change in almost 20 years. The fact that it retrospectively happened to be almost identical to industry standards such as IEEE 1233, means that these standards were born in proven engineering practice.

While being central to the management of requirements and to business analysis, this process is quite unknown in the IT departments where I have taught it in the last 15 years. This said, the individuals who take this course generally understand this process once they see it, the reasons behind it and the value it could bring them. They often say that they consider that this is the way they and their organization should work.

They are less sure of its applicability in their context. They offer various reasons for this: “my boss will not give me the resources; IT is not ready for it; we need a tool to implement it; I will not be given the means to do it.”

Tool support conjoint with organizational understanding and motivation is crucial for the long-term implementation of this process in practice.

The PizzaMat exercise is difficult to understand from a service requirement perspective. Not because of the subject but probably because of the complexity of service management. Service management seems to offer a view that is too generic for most business analysts who work in an organizational silo and must specify mostly functional requirements for a given application. Reasoning about the service provision, SLA and organizational context are beyond their scope and assigned responsibility.

Writing this paper made me realize that changes to the course were mainly driven by the changing industry landscape, vendor proposed methods and tools such as RUP and Rational Rose[3], as well as clients’ interest in these changes. Typically, the most recent impacts were triggered by customers’ adoption of ITIL, BABOK and Scrum. These influenced the course and my practice

[3] I could have also cited Microsoft’s Team Foundation Server ; HP’s Project, Service Management and testing tool suites; SAP’s ASAP; IBM Rational and others.

much more than what I learned from requirements engineering research, probably because they were of more immediate interest for my customers and students.

5 Outlook

Requirements management seems to be a hype cycle punctuated by keywords such as CASE, RAD and RUP. None of these are used anymore in the organizations I know. But the practice of Business Analysis is rising and, at least momentarily, is bringing back the concept of requirements. Indeed, the requirements engineering process is central to Business Analysis as described in the Business Analysis Body of Knowledge (BABOK). As a professional convinced of the importance of requirements engineering, my hope is that the mounting interest in Business Analysis will result in the possibility of implementing this process in real organizations.

References

1. Boehm, B., A spiral model of software development and enhancement. SIGSOFT Softw. Eng. Notes 11, 4 (August 1986), 14-24.
2. Martin, J., Rapid Application Development, Macmillan, 1991.
3. Jacobson, I, Christerson, M, Jonsson, P. and Overgaard, G, Object-Oriented Software Engineering A Use Case Driven Approach, Addison-Wesley, 1992.
4. Ernst & Young, Ernst & Young Navigator Systems Series Release 4.0, Overview Monograph, 1998.