An Agile Methodology for IHE-Enabled Deployments

Bruno Alves and Michael Schumacher

University of Applied Sciences Western Switzerland, 3960 Sierre, Switzerland
bruno.alves@hevs.ch

Abstract. Recent history of healthcare software is littered with failures, mostly attributable to bad design and the inability to capture user community's needs and workflows[1]. This paper presents a new agile methodology aiming at improving engineering practises in Integrating the Healthcare Enterprise (IHE)-based projects in small-to-medium scale hospitals and healthcare organizations. The work described here is a compilation of experiences in the field and is based on the successfull Scrum methodology, centered on IHE specifications that aims at providing value by making IHE developments easier and more predictable.

Keywords: Integrating the Healthcare Enterprise, IHE, security, methodology, planning, Scrum, interoperability.

1 From Hailing to Failing

The Therac-25 debacle is certainly the most serious case of computer-related accident to date involving patient death. A study published in [2] pointed to severe deficiencies in the engineering process. Ross Koppel revealed in an article published in 2005 [3] that *computerized physician order entry systems* (CPOE) can unexpectly and contrary to conventional belief increase the number of medication errors, because they fail at capturing user requirements and because *"machine rules do not correspond to work organization or usual behaviors"*.

Often, reponsibility for errors is not only attributable to a bad design, but also partly to the users, who are *"unlikely to accept blame for their own error or acknowledge their own inadequacies with respect to using the system*[1]. In 1999, Kohn and his colleages suggested in *To Err Is Human: Building a Safer Health Care System* [4] that healthcare professionals should increase awareness on how information technology could be applied to deliver safer care. Bad design, user misinformation and bad practises along with uncontrolled system growth and fast moving technology can actually lead to a new range of unexpected consequences. Awareness to the value of healthcare information exchange and interoperability[5] is probably the key to building better and safer systems.

P. Kostkova, M. Szomszor, and D. Fowler (Eds.): eHealth 2011, LNICST 91, pp. 17–25, 2012.

Integrating the Healthcare Enterprise (IHE)[1] is an initiative led by professionals and the industry to facilitate information sharing in healthcare. IHE promotes the coordinated use of well established standards (such as DICOM[2] or HL7[3]) to address particular clinical needs in specifications known as Profiles described both in terms of actors and transactions in a Technical Framework[6]. Each Profile solves an unique clinical use-case (document sharing, patient management, ...), but can be integrated in more general framework. Profiles are systematic implementation guides that allow developers to focus more on the actual user experience and less on the integration problems, leading thus to less frustration for the developers and more satisfaction to the users. The amount of available Profiles however and the high number of elements to care about may be a bit overwhelming as more Profiles are integrated into one product. The problem can be largely mitigated with a careful and methodic development approach.

Healthcare software has usually a rigid set of fixed requirements and one may think development in healthcare does not require the agility that is essential in more business-oriented software. Healthcare IT requires high connectivity, a great number of overlapping subsystems and presents challenging integration issues, difficult to foresee. Development teams are expected to be extremely reactive to changes and may have to respond to notable modifications in the environment variables (timeframe, team, ...). Agile methods provide the kind of flexibility necessary to handle these cases, and especially Scrum.

Scrum [7] aims at delivering as much quality software as possible in small frames of time. It improves on existing engineering practises by involving frequent management activities tracking any deficiencies or impediments in the development process. Scrum particularly adapted to reduced-size regional organizations, because its process does not require large teams. And, since the customer is usually the organization itself, the proximity allows for a better communication and helps in better serving its needs.

The IheTools project is a follow up of the Medicoordination project published in [8],[9] and [10]. Its main objective is to take the exploration of IHE Profiles started with Medicoordination a step further. This exploratory research is necessary to gain enough knowledge so to be able to assist regional hospitals and healthcare organizations in their IHE-related tasks.

This paper presents a new methodology based on the successful Scrum that aims at assisting small-to-medium scale regional hospitals and healthcare organizations in their IHE-related development tasks, by providing a consistent and systematic approach on building up frameworks based on IHE Profiles. A motivation case will be first introduced; the methodology will be then presented and discussed at the end of this work.

[1] IHE International, http://www.ihe.net
[2] DICOM Homepage, http://medical.nema.org/
[3] Health Level Seven International,http://www.hl7.org/

2 Motivation Case Study

2.1 On the Importance of a Good Plan

Our experience with IHE technologies was mostly acquired in the context of Medicoordination project. Although the project was successful in several aspects and the experiment was positively evaluated, the first prototype suffered from a few important design flaws that could have been avoided through a more careful design. This experience emphasized the importance of having good planning and a good methodology. Here are some experiences from the field:

As we came to learn, security is a topic that needs planning. It is in general a good practice to consider it already in the early stages of the product's development lifecyle. Although it may sound like a good idea to first develop an unsecured version and then add security to it, it is not. Security planning has many implications on the very foundations of the architecture and can lead to the creation of new unplanned breaches. IHE solves the problem partly, by providing a methodic and well documented security section in every Profile description. Common security concerns and elements to care about are well documented.

Furthermore, because healthcare software is usually provisionned with patient-centric sensitive data, not having a risk assessment and mitigation plan is unacceptable. Without it, the risk implications are not well understood, users not trained at is consequences and the final product may go well against local policies, against law or even against ethics.

Finally a mistake commonly found in research projects is to believe that a product will integrate flawlessly in the target operating environment. People often tend to believe there's no need to take the environment in consideration, even more in the early stages of the development. Murphy's Law[4] states that *"if something can go wrong, it will eventually go wrong"*. Taking the operating environment into account in the planning allows avoiding most nasty unforseen consequences.

2.2 Price Is What You Pay. Value Is What You Get[5]

Small-to-medium scale regional hospitals and healthcare organizations often have limited financial resources and assets. Although Return on Investment (ROI) is an important metric in decision making, the primary focus is on the value. Agile methods concentrate on the value by delivering functional iterations in short periods of time and also by placing the customer to the center of the discussion.

IHE can also provide value. A set of Open-Source IHE Profile implementations and frameworks was evaluated and a state of the art was published in [11]. All the selected implementations are free and IHE-compliant (most of them provide *compliance statements*) and may prove of good value for enterprises not willing to spend millions of euros in commercial licenses.

[4] Murphy's Law, http://en.wikipedia.org/wiki/Murphy's_law/
[5] Warrent Buffet 1930, American Investment Entrepreneur.

2.3 The Waterfall Falls, The Scrum Scrums

Unlike the traditional waterfall development model, agile methods do not assume that the requirements will not change. On the contrary, they embrace change as a mean to constantly improve the software architecture. Scrum was first introduced

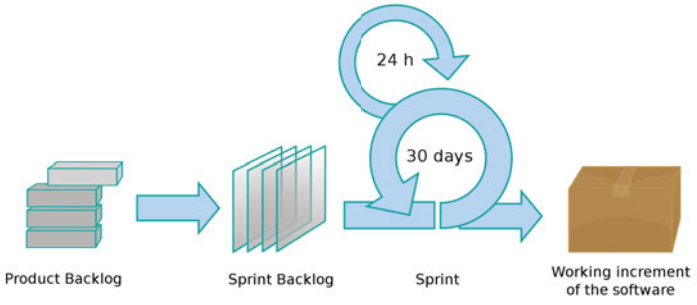

Fig. 1. The *Scrum* methodology process

in 1986 by Hirotaka Takeuchi and Ikujiro Nonaka [12] as a new approch to commercial product development. The primary objective was to increase the speed and flexibility of developments, by enforcing a single team with cross-functional skills to the whole process.

The Scrum methodology contains a set of predefined roles:

1. the "**ScrumMaster**" makes sure the process goes as intended
2. the "**Product Owner**" represents the customer
3. the "**Team**" works on the delivery of the product

The Scrum methodology does not make assumptions on the model used for the implementation, but rathers limits itself to the description of how to drive the development. Scrum consists in three main phases: pre-game, development and the postgame.

The *pre-game phase* starts with a meeting to define a set or prioritized high-level requirements, called *product backlog list*. A thorough planning is achieved, including risk assessment, security consideration, project team and so on. In a second step, an architecture model is designed from the backlog items and problems, which may arise are identified and mitigated.

The *development phase* undergoes successive iterations, called *sprints* that can last up to 30 days until the release is ready for distribution. Each *sprint* starts with a meeting in which several items are taken from *product backlog list* to work on. Items are often splitted in smaller tasks and inserted into the *sprint backlog*. Some environment variables, such as timeframe, quality, resources and so on are constantly re-evaluated. The Scrum Master is tied to management activities to assess that no impediment will ever affect the ongoing development.

The *post-game phase* leads to a fully working release. The development enters this phase as soon as all the environment variables (requirements) are completed. The system is ready to be released and preparation includes tasks such as integration, user training and documentation.

3 Description of the IHE-Agile Methodology

This section describes the three phases of the Scrum methodology including elements from the IHE documentation.

3.1 Preparing the Game

The pre-game phase contains all the required groundwork before the actual development can start. The phase starts with a planning, in which the central element is the generation of items in the *product backlog* by the members of the whole team, including customers, sales and marketing division, software developers and a few other roles, that may vary from one project to the other. The initial planning generally includes elements from security, risk assessment, team foundation, training needs, integration and testing plans and many others that have been detailed in other works on agile development.

IHE documents state that a risk assessment and mitigation plan should be written for each profile. IHE provides a Security Considerations section for every Profile, which is based on the mitigations identified in each risk assessment. The Security Considerations section is not a thorough standalone security assessment, but just deals with issues specifically relevant to interoperability. There is no precooked recipe for a risk assessment and mitigation plan, since risks usually vary from one project to the other. However, elements of the method used to write the Security Considerations section, which is described in the *IHE Cookbook: Preparing the IHE Profile Security Section*[13] can be used to help writing the global risk assessment plan. The method consists in identifying lists of risks, by imagining different scenarios and then assess their level of impact and probability of occurrence. Finally, mitigation of relevant risks for each profile is proposed.

Ethical and legal considerations need to be accounted for in the early stages of the development process. Ethical issues have to be well understood and communicated to the future users of the system. It should be clearly stated what is going to be done with sensitive data, who has access to it, what are the mechanisms to protect the data and what should be done in case of unauthorized disclosure. Legal aspects need to be analyzed transversely taking into account the laws of the country as well as the laws and policies of the lower levels (region, organization or business unit).

Security planning should be also partly done in the early phases of the development and improved during the subsequent *sprints* as the technical requirements become more and more clear. Some risk mitigation solutions proposed in the Technical Framework, must be analyzed and described in the security planning. IHE proposes some good articles on Profile-centric access control mechanisms[14]

and on the management of security and privacy. The *Template for XDS Affinity Domain Deployment Planning Handbook*[15] is usually used in planning the deployment of a XDS domain, but can also be used to plan organizational, operational and membership rules, as well as patient privacy and consent matters. System developers should not forget to attach the audit logging planning to the security section.

3.2 The Scrum Process

The subsections below describe the different phases of a single *sprint*. This is where the process shows agility. Indeed, in each *sprint*, environmental variables, such as requirements, technology and timeframes, are tracked and the development adapted if necessary.

Requirements. During the *Sprint Planning Meeting* punctuating the beginning of the next *sprint*, the Scrum Master, customers, users, Product Owner and Scrum team decide upon the goals and functionality. A second phase of the meeting gathers the Scrum Master and Scrum Team to decide how the product increment is going to be implemented and integrated into the existing product.

Analysis. The analysis phase includes elements such as security planning, usability and testing procedures.

Risk assessment must typically be done for every profile integrated in the final product, because different profiles generate different risks. Furthermore integrating with other Profiles generates even more risks.

Security planning must include all important information to help making security decisions. The team should be aware of topics such as actors, roles, authorizations, secure protocols, privacy, confidentiality and patient consent (involvement). It is important to define here what type of data is stored, where it is stored, who as access to it, for how long. This section may also contain emergency and bypass protocols as well as audit logs format and auditing placement.

Design. A typical IHE design should include low level aspects, such as IHE actors, transactions as well as security, interoperability annotations, communication protocols, encryption standards, protection mechanisms, role-based access controlling schemes and so on. All the links between the actors sould be clearly identified and annotated with the type of security they require (mutual TLS over HTTP, for example) and with used protocols at both ends. For example, if one link is used to send data from a system that produces HL7 CDA R2 documents, it is a good idea to write HL7 CDA R2 next to the source actor. This kind of annotations helps making sure that two integrated systems are talking the same language and the same version.

Implementation. Demos are important, because they help them assessing the progress of an ongoing development. The implementation process may consist in writing modules or deploying existing frameworks, but the general rule is that implementation efforts should always result in output, which is visible and presentable at the end of the *sprint*. Customers do not live on promises, they want value. IHE projects often tend to mix both coding and deployments. It is thus important to limit the scope of the current *backlog item* in order to produce some visible output until the end of the next deadline, even if that means to split the item for the next iterations.

Testing. Unit testing is a great tool for assessing the correct behaviour of the iteration, even though it is far from sufficient. There are other key aspects that must be tested and validated such as: interoperability, security and usability.

Interoperability testing comes down to assessing that your system is using the right communication protocols and standard. Sometimes, using a different version of a same protocol, for example HL7 v2.5 over HL7 2.4, may invalidate your efforts. Interoperability validation ensures that messages sent by your piece of software are well understood by the surroundings. IHE Profiles inform about the expected outcome of a particular transaction. The software tests must use this data to make sure the current products' iteration behaves as expected.

Furthermore, IHE Profiles rely heavily on proven standards and are generally interoperable. It may happen, however, that some specific features or options are not implemented in one particular IHE Profile implementation. Hence, it is necessary to perform interoperability validation before proceeding to the next phase. There exist some IHE testing frameworks, such as the NIST XDS Test Suite[6] or the now famous IHE Connectathons[7], where teams can test their IHE-enabled products against others and possibly receive compliance statements.

Security testing is a delicate matter, since in most small projects, there is no *true* security expert. Testing procedures must assess the security of the software module for scenarios designed in the planning phase. For safety critical systems, availability has also to be tested, by putting systems under heavy charges and inserting deficient nodes.

Usability is one aspect that can make or destroy your project. It is important to involve user judgment early in the development lifecyle. IHE Profiles are designed to integrate user's workflow without much disturbances.

3.3 The Game's Isn't over Until It's over[8]

The post-game phase represents the end of a release. The project enters in the post phase upon agreement that all environment variables are completed. The system is typically finished and release preparation tasks such as integration and delivery are done at this step.

[6] http://ihexds.nist.gov/

[7] http://gazelle.ihe.net/content/ihe-europe-2011-connectathon

[8] Yogi Berra, American professional baseball player and manager.

Integration. Integration is the operation, which consists in inserting the current release of the product in the operating environment. For implementations of typical IHE use-cases, IHE simplifies this integration, by providing guides and illustrations. Part of the integration work is about configuring the environment to line up with product's final specifications. Even through everything was planned in the pre-game phase, unexpected conditions may arise and a return in another *sprint* cycle may be necessary. The integration phase also includes extensive testing and quality assurance.

Delivery. At this point the product is ready to be delivered to the customer. In this phase, an extensive documentation has to be created and information procedures set up, including user training.

Postmortem: Lessons Learned. In almost all developments, some processes may go straight and others just don't. Recording in a written and accessible form what went wrong and what went right allows making continous process improvements. The recording usually happens in the *Sprint Retrospecive* meeting. Knowing what can adversely affect an ongoing development effort and how to respond to it, helps making future developments more predictable and comfortable for the team.

4 Discussion

The methodology presented above builds on the success of agile methodologies and on the rigor and consistency of IHE Profiles. It simplifies the development of projects in small-to-medium scale regional hospitals and healthcare organizations, by providing a guide about subjects that require a special care. Furthermore, this methodology also provides links to helpful IHE documents facilitating the process of writing down good risk assessement plans, creating SOA architectures based on IHE Profiles or preparing the security of the product.

This methodology is meant to be used as a supplementary tool and not as a all-in-one guide. Is was created as a response to problems we had in our previous developments. No standard evaluation was performed using existing frameworks, which may restrain the scope of usability of this methodology to non-safety critical and non-business critical projects . However, concepts presented here are taken from our past experiences and may still prove valuable in order to avoid the same mistakes again and again.

5 Conclusion

We presented here a new agile methodology centered on IHE Profiles that allows assisting small-to-medium scale healthcare development tasks based on IHE. The methodology presented here builds on the agility principles of the successful Scrum agile method and is primarily intended to be used by small-to-medium scale regional hospitals and healthcare organizations where teams are sufficiently small and proximity is good.

References

1. Laplante, P.A., Nell, C.J., Sangwan, R.S.: Healthcare Professionals' Perceptions of Medical Software and What to Do About it. J. IEEE Computer 39, 26–32 (2006)
2. Leverson, G.S., Turner, C.S.: An Investigation on the Therac-25 Accidents. Computer, 18–41 (1993)
3. Koppel, R., et al.: Role of Computerized Physicain Order Entry Systems in Facilitating Medication Errors. JAMA 9, 1197–1202 (2005)
4. Kohn, L.T., Corrigan, J.M., Donaldson, M.S.: To Err Is Human: Building a Safer Health System, p. 80. Nat'l. Academies Press (2000)
5. Walker, J., Pan, E., Johnston, D., Adler-Milstein, J., Bates, D.W., Middleton, B.: The value of health care information exchange and interoperability. J. Health Affairs, hlthaff–w5 (2005)
6. IHE IT Infrastructure (ITI) Technical Framework. IHE International, vol. 1–3, Revision 7 (2010)
7. Schwaber, K., Beedle, M.: Agile software development with Scrum. Prentice Hall PTR (2001)
8. Alves, B., Müller, H., Schumacher, M., Godel, D., Abou Khaled, O.: Interoperability prototype between hospitals and general practitioners in Switzerland. In: Medinfo 2010, Cape Town, South Africa, pp. 366–370. IOS Press (2010)
9. Müller, H., Schumacher, M., Godel, D., Abou Khaled, O., Mooser, F., Ding, S.: MediCoordination: A practical approach to interoperability in the Swiss health system. In: The Medical Informatics Europe Conference (MIE 2009) (2009)
10. Alves, B., Schumacher, M., Godel D., Richard, P. Abu Khaled, O., Müller, H.: Prototype d'interoprabilit entre hpitaux et mdecins traitants. In: Actes de GISEH 2010, Conference Francophone "Gestion et Ingnierie des Systmes Hospitaliers", Clermont-Ferrand, France (2010)
11. Alves, B., Schumacher, M.: The Open-Source Integrating the Healthcare Enterprise (IHE). In: Proceedings of the IADIS e-Health Conference, Rome, Italy (2011)
12. Takeuchi, H., Nonaka, I.: The new new product development game. J. Harvard Business Review 64, 137–146 (1986)
13. IHE Cookbook: Preparing the IHE Profile Security. IHE International (2008)
14. Caumanns, J., Kuhlisch, R., Pfaff, O., Rode, O.: IHE ITI Infrastructure White Paper: Access Control. IHE International (2009)
15. IHE IT Infrastructure Technical Commitee White Paper: Template for XDS Affinity Domain Deployment Planning. IHE International, Version 15 (2008)