HOW DO XP, SCRUM AND ASD BUILD THE RIGHT SOFTWARE?

Angela Martin, Robert Biddle, James Noble
Victoria University of Wellington, New Zealand
angela at mcs.vuw.ac.nz

This position paper explores a particular area within software development: requirements analysis and management. We specifically focus on only three agile methods, XP, Scrum and ASD. These methods all highlight, in their respective books [1,2,3], the requirements difficulties often encountered in traditional development. The next sections cover how XP, Scrum and ASD address these difficulties.

EXTREME PROGRAMMING. There are no new practices in XP, it simply takes existing practices and turns them up to an extreme level. The revolution is making the practices reinforce each other. XP faces requirements difficulties head-on by recommending:

(a) The customer is an integral part of the team and should be on-site with the team
(b) The customer writes user stories and then discusses each requirement directly with the programmers
(c) The customer is responsible for *all* business decisions including prioritising user story development
(d) The small 2-3 week iterations allow the user to evolve their requirements based on concrete working software
(e) The customer regularly tests the software to confirm it works as expected

SCRUM. Scrum is a project management method; it aims to provide a mechanism to control the chaos surrounding the communications between the business and the project team. Scrum streamlines the requirements communication:

(a) There is one central prioritised list of all requirements, the *product backlog*
(b) The product backlog is controlled by one person, the *product owner* , however many people contribute to the list
(c) The 30 day sprint allows the project team to focus on a frozen set of requirements, while allowing the business to change their requirements after a tolerable delay
(d) The software is reviewed at the end of each sprint to ensure ongoing feedback to the project team and the business

ADAPTIVE SOFTWARE DEVELOPMENT. ASD is based on complex adaptive theory and describes the mind set and principles required to succeed in developing software iteratively and incrementally. ASD recommends the following for requirements:

(a) There are a set of artefacts (project vision, data sheet and specification) that ensure a shared project vision exists.
(b) Collaboration techniques are used to evolve requirements: JAD sessions, Customer Focus Groups and finally post-mortems or process improvement reviews.
(c) A *collaboration facilitator* role is introduced to focus on "thinking" about and planning collaboration rather than simply "letting it happen".

SO, HOW DIFFERENT ARE THEY? There is significant cross-over between the methods, including:

- The product backlog, user stories and product specifications all ensure there is a centralised list of requirements
- The on-site customer role, the product owner role and the JAD/CFG sessions all aim to ensure the developers do not need to deal with conflicting requirements, including the prioritisation of requirements
- The small and regular iterations ensure there is opportunity to learn and to evolve requirements.

So, while there is some difference in the details of the methods, many of the practices could potentially be interchanged. One of the key overall differences between XP and Scrum is their emphasis on different aspects of software development; XP emphasises programming, while Scrum emphasises project management. XP provides reinforcing practices for developers and Scrum provides comprehensive management practices for project managers. ASD concentrates on outlining learning and collaborative techniques and theories for all project members, and introduces a collaborative facilitator role that specialises in collaborative and learning based communication, including the communication between customers and developers. Both XP and Scrum appear to under-emphasise the challenging job [4] facing the *on-site customer and product owner* and in gathering and prioritising the requirements for the project. ASD begins to address this issue by recognising the need for structured collaborative techniques and roles in this area.

REFERENCES

1. Beck, K. *eXtreme Programming Explained: Embrace Change*, Addison Wesley, 2000.
2. Schwaber, K., Beedle, M. *Agile Software Development with Scrum*, Prentice Hall, 2001.
3. Highsmith, J. Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, Dorset House Publishing, 2000.
4. Martin, A., Noble, J., and Biddle, R. *Proceedings of the Fourth International Conference on eXtreme Programming and Agile Processes in Software Engineering*, Giancarlo Succi (Ed.), "Being Jane Malkovich: a Look into the World of an XP Customer". Lecture Notes in Computer Science 2675, Springer-Verlag. 2003.