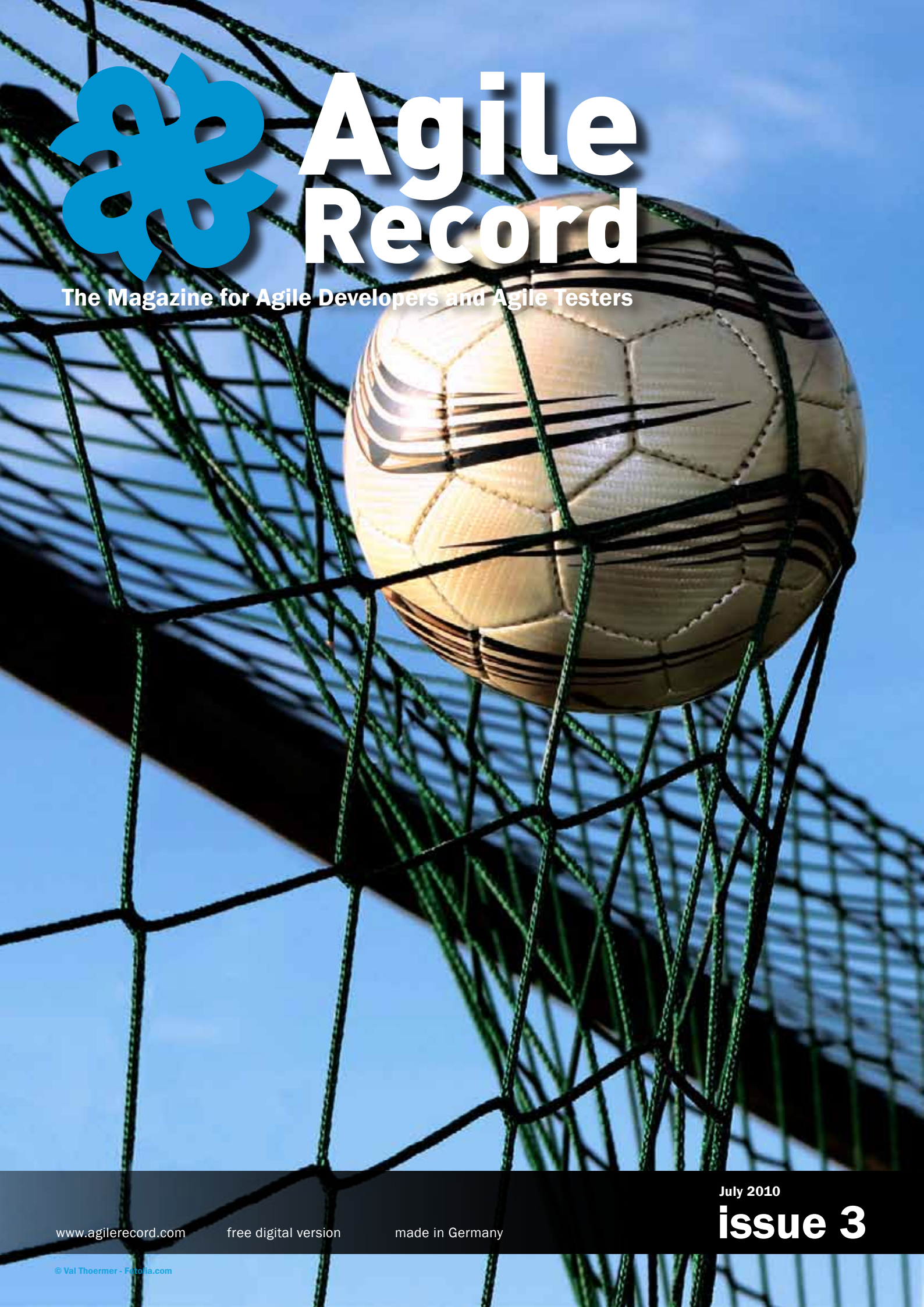

Agile Record

The Magazine for Agile Developers and Agile Testers

July 2010
issue 3

www.agilerecord.com free digital version made in Germany

SCRUM & Testing: Assessing the risks

by Erik van Veenendaal

Food for thought

One of the Agile methods currently being applied in many organizations is SCRUM, a method for project management that uses incremental work cycles known as sprints to provide frequent opportunities to assess and revise direction. As so often, however, it is simple but not easy! The SCRUM method doesn't say much about testing, and in most books only unit testing and user acceptance testing are addressed. Have integration and system testing now become obsolete? Developers and business representatives should do the major part of the testing activities within Agile developments. It seems like utopia, but what does it look like in real-life? Exploratory testing is often identified as the main technique; what about our traditional test design techniques - are they no longer needed? These and many more questions need to be addressed when moving from a traditional V-model type development methodology to an Agile methodology, such as SCRUM. When reading this column, you may get the impression that I'm against SCRUM, Agile development and the like. You could not be more wrong! However, I would like to point out some testing issues from practice that one encounters when starting to apply SCRUM. In other words: "food for thought" for the reader when moving from a traditional development methodology to Agile development/SCRUM. A checklist of risks that should be mitigated or at least considered during the change process. Note that most of the risk items are not even new, but were already a challenge during traditional V-model development and testing. By the way, my personal experiences are not from the games and internet industry, like many simplified stories one hears at conferences, but from more critical environments such as finance, embedded software and medical systems, where quality matters and is a key business driver.

Risk item 1: Test professionals

One needs to assess the current set of testers and decide whether they are "fit for Agile".

Within Agile testing, quality is the team's responsibility. The test analysts now have a different role. In addition to testing, they have to coach business representatives and developers on testing issues, review unit tests, etc. The tester is also involved in estimation sessions, defining the exit criteria (definition-of-done), reviewing stories and making them testable. All of this requires a senior person with good communication skills. It's almost like we require ISTQB Advanced Level testers; people that understand and are able to explain technical testing, business testing, test design and test management issues. What do we do with our junior testers? What do we do if we don't have senior testers with the right skills? Lisa Crispin in her book "Agile Testing" defines a number of qualities that agile testers should possess: deliver value to customers, have courage, respond to change, continuous feedback, etc. If you study those in more detail, there is little to no difference to what I would call a senior test analyst (in a traditional environment). If we analyse things, Lisa Crispin says that we need real professional senior testers with the right knowledge, skills and mindset. So what is new? Doesn't this sound familiar?

Risk item 2: Developers

Within Agile software development, unit testing is essential. Why is it essential only in Agile, and why does it now get all the focus? It should have been essential in every lifecycle methodology! Can developers all of a sudden apply test design techniques? Most of them have never had any training in structured testing, not even at foundation level. Unit testing is not as easy as it seems, understanding test design is something not so common for most developers. Writing test code that makes the code fail as in XP does provide structural coverage, but not a high level of functional coverage. Getting developers into testing needs a practical approach. We at Improve Quality Services often provide workshops with hands-on real-life practical cases to get them started. Test automation, essential with any incremental development methodology, is also needed. As test professionals have already

known for years, this is still a big challenge for most organizations for various reasons. All Agile is saying is, we need high quality unit tests and fully automated regression tests. Things that have never been easy in the past. So what is new? Doesn't this sound familiar?

Risk 3: Test levels and test types

Most books on Agile (testing) have a strong focus on unit testing and user acceptance testing.

Whatever happened to good old integration and system testing? What about system integration testing in a systems-of-systems environment? Of course, there isn't just one answer, since these terms mean different things in different organizations. The main problem I have with this approach is that again, after all these years, we have to re-explain to management why integration testing and system testing are needed. Caper Jones has already taught us that unit testing doesn't get far beyond a Defect Detection Percentage (DDP) of 35% to 40%, and together with only validation-oriented use cases used in user acceptance testing, just doesn't get the job done! Thorough test cases using, for example, decision tables or classification trees are still required in many instances. Also, exploratory testing has its limitations (sorry, James, Michael). Some test levels or test types take place outside a sprint, which seems impossible according to the Agile theory, but this is what I see successfully happening in practice. Reliability and performance testing for example are often test types that run for more than four weeks, and organizing an operational acceptance test or a system integration test within a sprint has been shown to be very difficult in practice. As a result, in practice some of the testing often takes place outside the SCRUM sprint. A major issue is then to align the testing efforts and approaches inside and outside the sprint. What needs to be done? Establish a test strategy that covers all testing activities (levels and types). To unambiguously define testing is usually a huge step in the right direction. This has been the issue in the past with the V-model and is also needed for Agile software development. So what is new? Doesn't this sound familiar?

Many more challenges

The issues raised above are by no means the full risk list. I wanted to prioritize, keep it short and "write Agile". Some other risk items to be considered:

- **Risk item 5: The Test Manager**
  What happens to the test manager whose role is now obsolete?
- **Risk item 6: Stress**
  If we deliver software every four weeks, there will be stressful release periods more often. What does this mean for those involved?
- **Risk item 7: Distributed teams**
  How do we organize Agile testing with distributed teams, and how does it fit with outsourcing?
- **Risk item 8: Management understanding**
  Does management really understand the background to the Agile manifesto? Even within SCRUM, there are limitations to change!
- **Risk item 9: Business involvement**
  How easy is it to get the right business representatives onboard and get them to do some testing? (In my experience, even reviewing test cases and providing input to a product risk analysis is often asking too much.)
- **Etc.**

The manifestos

Again, I would like to emphasize I'm not against SCRUM. I'm actually a big fan of team-based working and Agile. In practice, however, it just isn't as easy as explained in the books and presented by some of the so-called gurus. If you are able to handle some or all of the risks presented, SCRUM projects have shown to be very successful. It usually helps enormously to have a detailed discussion about the Agile manifesto with all those involved. What does it really mean? It's all about the philosophy behind the manifesto, understanding what is meant, and not about the actual wordings. Someone who really understands the Agile manifesto may even notice that there are a lot of similarities with the test process improvement manifesto [Testing Experience, Issue 4, 2008].

- *Flexibility* over detailed processes
- *Best practices* over templates
- *Deployment orientation* over process orientation
- *Peer reviews* over Quality Assurance (departments)
- *Business-driven* over model-driven

Most of these statements can easily be adapted to Agile software development. So what is really new? I "even" have a number of clients at TMMi levels 2 or 3, that practice SCRUM. In fact, those organizations that had the structured testing in place and/or were at CMMI level X seem to be more successful at SCRUM than others. Again "food for thought" ...

I hope this short paper makes you aware of some of the testing issues that come with the implementation of SCRUM in an organization or project. Discuss them with team members inside and outside testing and management, and find practical result-driven solutions. Good luck in mastering SCRUM. ■

> About the author

Erik van Veenendaal *(www.erikvanveenendaal.nl) is a leading international consultant and trainer, and a recognized expert in the area of software testing and quality management. He is the founder of Improve Quality Services BV (www.improveqs.nl). At EuroStar 1999, 2002 and 2005, he was awarded the best tutorial presentation. In 2007 he received the European Testing Excellence Award for his contribution to the testing profession over the years. He has been working as a test manager and consultant in various domains for more than 20 years.*

He has written numerous papers and a number of books, including "The Testing Practitioner", "ISTQB Foundations of Software Testing" and "Testing according to TMap". Erik is also a former part-time senior lecturer at the Eindhoven University of Technology, vice-president of the International Software Testing Qualifications Board (2005–2009) and currently vice chair of the TMMi Foundation.