Whitepaper

sqs.com

Offshore Agile Testing

Bridging the Gap between Craftsmanship and Industrialisation

Author: Shrikant Dhamal
Principal Consultant
SQS India Infosystems Pvt. Ltd.

Published: August 2013

SHRIKANT DHAMAL
Principal Consultant
shrikant.dhamal@sqs.com

Shrikant Dhamal is Associate Director working with SQS since 2002. He holds CSM, PMP, CSTP, ISTQB, SCJP certificates and a degree in mechanical engineering. He has over 13 years of industry experience with the last 10 years in professional testing across a wide range of industry sectors. His key responsibilities include programme/project management, test management, test automation and managing a distributed team.

Contents

1. Management Summary . . . 3
2. Introduction . . . 4
3. Market – Current Status and Outlook . . . 7
4. Offshore Agile Testing . . . 8
   4.1. Offshore Readiness . . . 9
   4.2. Training . . . 11
   4.3. Communication . . . 12
   4.4. Governance . . . 14
   4.5. Automation . . . 16
5. Case Study . . . 18
6. Conclusion and Outlook . . . 19
7. Bibliographical References . . . 20

1. Management Summary

Agile Software Development and IT Offshoring are most likely two of the most prominent and prolific industry trends of the past months. Market research figures demonstrate that both trends have gained substantial adoption and will continue to gain momentum in the foreseeable future.

Both trends attempt to tackle similar root causes, albeit with dramatically different solutions: all organisations need to streamline their IT projects due to market pressure, trying to simultaneously boost speed and efficiency while increasing product quality and decreasing costs. Both Agile and Offshoring promise to support this optimisation.

Yet, Agile and Offshoring are very different indeed. Agile promotes small-scale planning and implementation work, combined with instilling an ethos of master craftsmanship into small cross-functional teams. In this model, knowledge is shared by intensive interpersonal communication. Offshoring, on the other hand, attempts to leverage differences in labour cost by moving whole work packages to an emerging market where employing staff is cheaper, thus relying on the ability to clearly define work packages and to transfer knowledge across geographical, cultural and language boundaries. Testing is often chosen for Offshoring as it is not seen as part of the organisations' core competencies.

Differences notwithstanding, both Agile and Offshoring are being adopted by organisations at the same time, requiring a resolution to the conceptual incompatibilities between them. Indeed, it is possible to successfully combine Offshoring and Agile for the test capabilities in IT projects.

The present paper demonstrates how the success factors of Offshore Readiness, Training, Communication, Governance, and Automation influence the course of testing and the project as a whole, and which measures to take to put projects on the road to success. The implementation of such an approach is shown with a case study of an actual Offshore Agile Testing project.

2. Introduction

The past months have seen the emergence of two main, enduring trends in the IT world. The first of these trends is the move towards new and improved software development approaches. This move has been fuelled by the insight that traditional software development approaches with process models such as the Waterfall Model or V-Model do little to support IT projects in coping with the complexity of modern IT systems. The result of this lack in support is a high failure rate of IT projects as indicated by the biannual CHAOS Report, furnished by The Standish Group (The Standish Group, 2009).

As a consequence, a number of existing approaches were bundled and extended under the moniker of 'Agile Software Development'. This set of approaches has remained strongly influenced by its grassroots past and as a consequence remains fluid: approaches, methods, and techniques are added to the roster or extended in order to adapt to new insights or a changing context. Thus, it is difficult to characterise Agile by describing its approaches or methods. However, all approaches have something in common, a set of common values which is written down in 'The Agile Manifesto' (see Figure 1).

The values stated in this manifesto profoundly impact on how Agile approaches solve the task of designing and implementing software. For instance, many Agile approaches insist on small, empowered and cross-functional teams. This means that traditional role models such as developers vs. testers have no place in Agile teams – every team member needs to be prepared to conduct or support any task. Team members are encouraged to take pride in what they do. In a way, an Agile team member is reminiscent of a skilled master craftsman who can handle all steps of producing a good by himself.

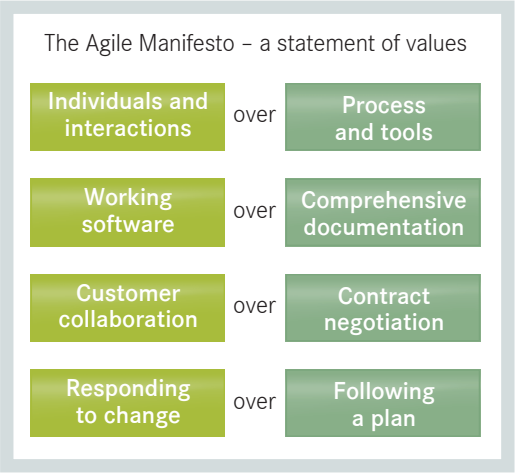

Figure 1: The Agile Manifesto (Tyler, 2010)

Additionally, working in small, recurring iterations is preferred over all-encompassing project plans that subdivide a project into distinct phases with quality gates, handovers and sign-offs of predefined deliverables. Instead, teams are encouraged to plan only as far ahead as is necessary to perform the next iteration, forcing everyone to break down huge and complex systems into manageable real life portions at the cost of possibly having to revisit decisions later on. This also allows teams to react to changes in real life requirements much more swiftly than a team working according to a classic approach: each iteration gives the system's stakeholders the opportunity to revisit their requirements and change the direction of the project.

Figure 2 details one of the most popular Agile approaches, the Scrum Framework. It provides a boilerplate process to make requirements to the system explicit, to break them down into manageable chunks, to facilitate the iterative implementation of these chunks, and to review the results to ensure that they

indeed do solve the user's problem. This is achieved by implementing a set of iterations. On a day-to-day basis the full team meets up in a daily Scrum meeting to discuss progress, impediments, and challenges in order to jointly resolve issues as soon as and as efficiently as possible. For a duration of one to four weeks, a sprint comprises the iteration necessary to implement the set of features that has been chosen in an elaborate planning meeting. At the end of each sprint, the team is expected to deliver a running system that can be inspected by stakeholders who in turn will give feedback.

The whole process is facilitated by a role called Scrum Master, but unlike classic team or project managers this Scrum Master only facilitates. The decision power over most issues remains with the team.

As a direct consequence of the Agile values, documentation is restricted to instances where it is deemed necessary – and, indeed, in many cases it is attempted to have the system document itself. For instance, with the Behavioural Driven Development (BDD) technique, the documentation of requirements or features is reused for specifying acceptance test cases, thus reducing the amount of documentation created. The same is true for all other forms of technical documentation. This differs strongly from classic approaches where often specific pieces of documentation (e.g. a system requirements specification, a system architecture specification) are considered formal deliverables without which a project cannot progress to the next phase of the life cycle.

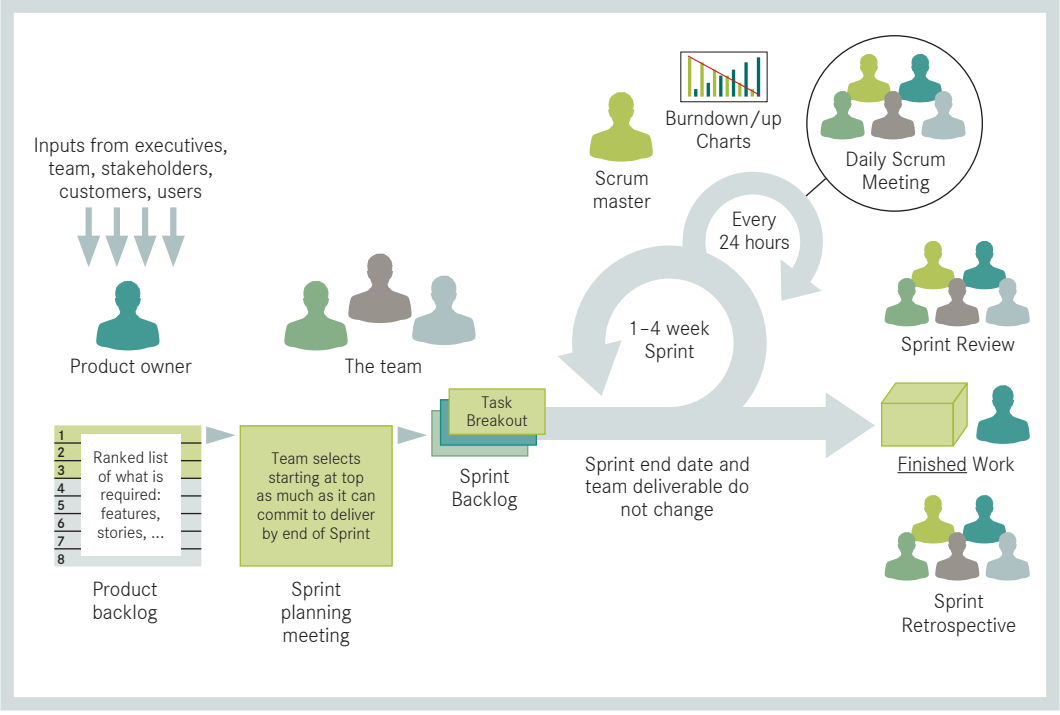

Figure 2: The Scrum Framework (Agile for All, 2008)

The second major trend in IT is the move towards an industrialisation of IT work, i.e. the move towards concentrating on core competencies. As a consequence, those work packages that are not considered a core competence, but still need to be handled, are awarded to outside agencies. In the case of outsourcing, the outside agency is a contractor; in the case of Offshoring, the outside agency resides in a different geographic location, often in an emerging market such as India or China. In both cases, the goal is to save costs and reduce direct risks by having a third party conduct this work. Offshoring is seen as a particularly attractive alternative, due to the fact that the dramatic difference in labour costs promises an equally dramatic reduction of project costs.

In the IT world, testing is often seen as such a work package: the owners of large-scale IT projects (such as banks or insurance companies) do not see testing as part of their portfolio of core competencies and, thus, attempt to have this work conducted by an offshore contractor.

It is easy to appreciate that owners and outside agencies may have conflicting viewpoints regarding many key aspects of software development:

- Awarding specific work packages to contractors requires planning and designing the full system in one go, while Agile approaches would favour working in small increments and many iterations.
- Similarly, industrialisation expects a clear division of roles and responsibilities to be in place. For instance, when outsourcing the testing function, it is expected that a defined number of roles is outsourced. Due to the cross-functional nature of an Agile team, this expectation is often difficult to fulfil.
- Outsourcing and Offshoring in particular create divisions between teams, due to organisational, cultural, language or geographic boundaries. Agile expects team members to be collocated and to be in continuous communication.
- Reference documentation that would exist in classic IT project life cycle models and which would help achieve the transition of responsibility from in-house to the contractor is often not available.

Yet, driven by the needs of simultaneously cutting costs, improving efficiency, increasing quality, and decreasing time to market, organisations are often embracing both trends at the same time. This results in situations where in-house teams are encouraged to adopt Agile approaches while parts of IT project work are contracted out or offshored. This situation poses the challenge of how to integrate two clearly conflicting approaches while preserving as many of the potential benefits as possible.

3. Market – Current Status and Outlook

As this whitepaper discusses the confluence of two distinct IT trends, it is useful to understand the market impact of each of these trends individually first. For Agile, Forrester Research have published figures indicating that by 2010 already 35% of all surveyed IT organisations reported that they were using Agile software development methods (Krill, 2010). When including non-Agile but iterative software development models such as the Rational Unified Process or Spiral Development, this figure increases to 46%.

This implies that at a first meeting with a new customer there is a one-in-three to one-in-two chance that this customer will have established an Agile or iterative software life cycle model. Agile truly has become a mainstream approach for IT projects.

For IT industrialisation, the picture is somewhat more heterogeneous. Offshoring and outsourcing have been around for a while and can be considered established approaches for the European and North American markets. However, the shares of the potential market are still comparatively small.

Figure 3 demonstrates the significance of the trend towards IT industrialisation: more and more companies use third parties to deliver IT services. This trend will increase over the next years. Market research firm PAC in a research analysis (Leclerque, 2010) expects an outsourcing ratio for 2020 of about 25% – this means that one out of four euros in the IT market will be spent on outsourcing.

Offshoring work is strongly connected to the concept of outsourcing, because in order to reap the highest benefits, organisations usually use a mixture of both: work packages are contracted to outsource suppliers who employ resources from offshore locations such as India to deliver.

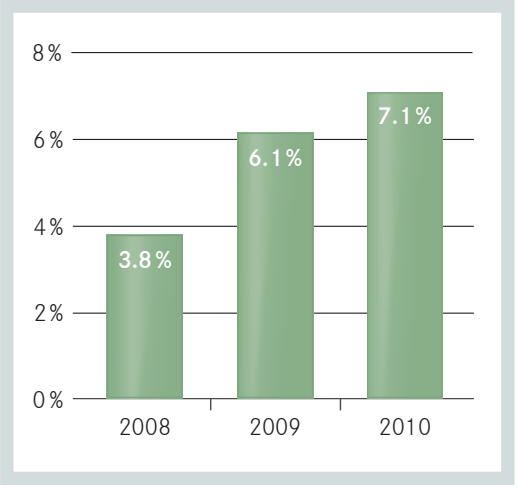

Figure 3: Outsourcing ratios 2008 to 2010 (Computer Economics, 2010)

4. Offshore Agile Testing

As a result of market demand forcing organisations to move towards outsourcing and Offshoring work while at the same time pressing them to embrace Agile approaches to software development, projects that operate in this space become subject to the strong and potentially destructive forces outlined in Section 2.

The challenges posed by factors such as the physical remoteness of dispersed teams need to be actively counteracted in order for Offshore Agile projects to succeed. In the case of Offshore Agile Testing, the following five criteria are crucial to project success:

- Offshore Readiness
- Training
- Communication
- Governance
- Automation

One of the paramount preconditions for project success is that all project parties, customers and suppliers alike, have achieved specific levels of **Offshore Readiness** so that they are truly in a position to send work packages offshore. For example, if the customer's organisation is not mature enough, it will be impossible to isolate work packages and to ensure that some sort of process is adhered to. As a consequence, Offshoring work will be extremely difficult since the frequent changes in requirements or processes will eliminate possible savings.

In the context of Offshore Agile Testing, **Training** denotes the means necessary to support all staff members that are not already familiar with the Agile approach to software development, in order to master the paradigm shift. This is especially true for offshore resources as they often cannot fully benefit from being able to interact with onsite staff members for whom Agile is part of their daily routine. It is training's function to ensure that the tenets of how the project is run are well understood by offshore and onsite team members alike, so that they can function as one team.

Communication is probably one of the key areas to ensure success when employing Agile methods – even more so in a setting where the team is not collocated but rather dispersed. This dispersal often not only means that team members are unable to regularly be in the same room, but also entails potential issues such as time difference as well as cultural or language barriers. While the disadvantage of not being able to be in the same office cannot be fully overcome, a number of methods and techniques, when used in a consistent fashion, can ease the disadvantage to some extent.

Fourthly, **Governance** and project management practices need to reflect the hybrid nature of Offshore Agile projects to be effective. For instance, it is imperative for all team members, onsite and offshore alike, to have full access to all relevant documents, e.g. files, metrics or dashboards. Additionally, some Agile principles cannot be adhered to in full due to the dispersal of team members. In many cases, more documentation is needed than would be deemed necessary in a regular Agile context.

Agile methods emphasise the use of **Automation,** where reasonable, to transfer non-core workload from Agile team members. This holds true for testing in particular, as can be witnessed by the large array

of tools that attempt to support the automation of developer or acceptance testing. Subsequently, as we are focusing on Offshore Agile Testing, test automation becomes a highly relevant topic, as Agile team members will expect the testing function to integrate with the other automated activities such as continuous integration or daily builds.

We will now look into these areas in more detail in order to identify the key success factors that need to be implemented in an Offshore Agile Testing context.

4.1. Offshore Readiness

It is an open secret that Offshoring is not the silver bullet that ends all pain the organisations in the Western Hemisphere might have. As the first long-term experiences with IT Offshoring come to light, so do best practices when it comes to deciding when to offshore, what to offshore, and to whom.

Understanding that both partners – the customer as well as the Offshoring partner – need to have achieved minimum levels of maturity is a key insight without which Offshoring is often heading for disaster. In fact, experience shows that in many cases Offshoring initiatives do not fail due to a lack in maturity of the Offshoring partner but rather due to the customer's immaturity (Simon & Simon, 2012). This holds true regardless of the concrete software development approach used, i.e. an evaluation of Offshore Readiness is required for both Agile and non-Agile projects alike.

Best practices have emerged which stipulate that a systematic decision on Offshoring has to be made before moving work packages offshore. This systematic decision needs to reflect the dimensions of Business Readiness and Offshore Readiness of all the systems or projects considered.

Figure 4 details the process used for such an assessment. In a first step, an up-to-date inventory of relevant applications, systems or projects is compiled. This step is required since in many cases maturity differs between organisational units' systems, so that for one project testing could be offshored while a different project might be so immature that Offshoring work successfully would be impossible.

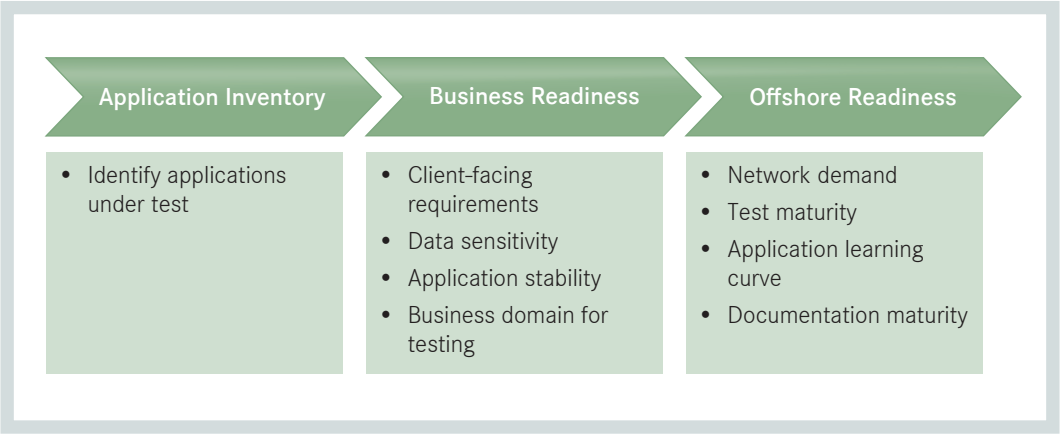

Figure 4: Assessment process for Offshore Readiness

After the first step, all relevant projects or systems are well known. Now, the two dimensions of Business and Offshore Readiness can be evaluated. Business Readiness is concerned with how well-prepared the business side is for any given system or project. This can be further refined into the criteria of how client-facing the testers need to be, whether sensitive (e.g. personal) data is involved, how stable the application already is, and how flexible the business domain expects the test organisation to be when it comes to handling short-term requests.

Similarly, the dimension of Offshore Readiness is concerned with the risks associated with Offshoring work on the specific system. This includes technical parameters such as the network demand, which governs how well remote testing will work, but also factors like the maturity of the existing test function, the complexity of the application, and how well it is documented.

These criteria are evaluated by conducting interviews with relevant stakeholders and by analysing available documentation such as process models, architecture documentation, requirements, or user manuals. Once all source information has been collated, a number of checklist items for each of the above-mentioned criteria are evaluated and assigned a score. The individual scores are finally aggregated into scores for the two dimensions of Business Readiness and Offshore Readiness.

Figure 5 shows an example of what such results would look like for a hypothetical customer. As can be seen, the results are subdivided into four quadrants. For the upper right-hand quadrant, both Business and Offshore Readiness are high. These are the systems/projects where Offshoring will be easiest and will provide the greatest benefits to the business.

The lower right-hand quadrant contains those systems that could technically be offshored quite well but where the business is not ready yet. In these cases, improvement programmes are set up to boost Business Readiness and continue with Offshoring afterwards. These systems constitute the second cluster to consider for Offshoring.

The upper left-hand cluster contains the systems revealing high Business Readiness but low Offshore Readiness. In those cases, it often is a more complex or expensive task to improve technical factors such as network bandwidth or documentation. Once these challenges have been resolved, the systems can be considered for Offshoring.

Lastly, the systems in the lower left-hand corner are lacking Business Readiness as well as Offshore Readiness. Offshoring them would not only be a recipe for disaster but would also deliver little value to the business. Any attempts to offshore them would need to be put on hold.

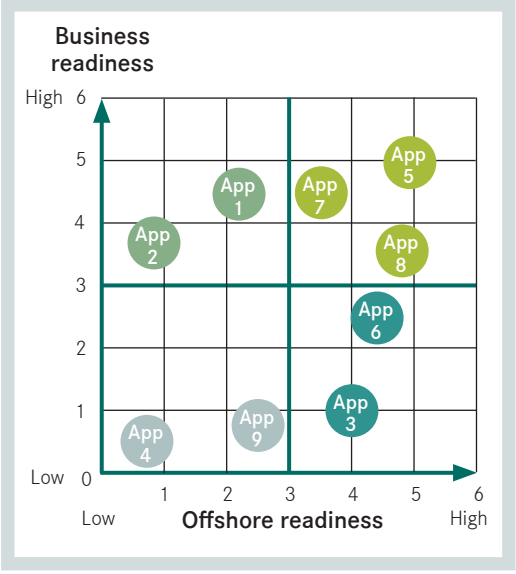

Figure 5: Sample of assessment results for Offshore Readiness

4.2. Training

Moving from a traditional, non-Agile methodology to an Agile methodology such as Scrum can be considered a dramatic shift of paradigm. The Agile Manifesto illustrates quite well how different fundamental values are considered in an Agile context (see Figure 1).

In many traditional projects – especially those following classic lifecycle models such as the Waterfall Model or V-Model – preferring individuals and interactions over processes and tools, or working software over documentation, would constitute an unheard-of sacrilege.

Therefore, team members experienced in non-Agile methodology are often sceptical about Agile methods in general, and methods such as Scrum in particular. Typical questions arising during such a transition are the following:

1. Is it possible to execute a project without a project manager?
2. Can a project be executed without defining specific job titles (e.g. tester, developer)?
3. Can a tester ask for support for testing from non-QA members, or can he support developers?
4. Can a tester actively take part in defining the project scope and its planning?
5. Can a team member make decisions without asking or obtaining approval of seniors?

One of the key success factors of implementing Agile methods is providing support for this change process, a key element of which is providing sufficient training for the team members. Each team member needs to understand how things work, so that they are convinced that Agile can operate successfully. The team members' trust is indispensable to successfully making the mental transition. In this situation, it can be helpful to have some more experienced members in the team, and to have access to Agile experts to mentor the team as needed.

From the methodological point of view, the training can be conducted in a number of ways:

- Classroom training including simulation of practical Agile scenarios
- Watching recordings of real-life Agile teams, such as authentic Scrum videos
- Practical 'lab' training sessions allowing for mistakes
- Q & A sessions with Agile experts whom team members have the opportunity to interview
- Continuous mentoring

The goal of conducting these trainings is to provide all team members with the necessary Agile skills. In particular, this involves understanding the benefits of Agile over non-Agile, especially the benefits of establishing a closer relationship with the business.

A key aspect that needs to be well understood is how testing is involved in all activities right from the inception, in order to provide early and continuous feedback about the quality of the product as compared to the business requirements. Here, testing is not the 'last resort' measure before shipping and, therefore, it is not conducted in isolation after development but rather simultaneously, and in collaboration with the developers.

Consequently, the following two aspects are imperative:

- Testers collaborate with the business or the product owner to define unambiguous requirements and acceptance criteria. When validating the requirements supplied, testers need to ensure that they are complete, unambiguous, consistent, testable, traceable, and modifiable by conducting critical reviews and asking the right questions.
- Testing is conducted at several layers and by all roles in the team. Developers will create unit testing suites to perform unit tests for all pieces of code that they are working on. Integration testing can be done to some extent using the continuous integration facilities available in Agile projects. And towards the end of a sprint, acceptance testing is a dedicated activity ensuring that what has been done complies with the specifications the customer provided as part of the stories. The acceptance criteria specified early on in the sprint serve as criteria to accept or reject pieces of work.

This requires a crucial attitude change in all team members: most, if not all, tasks in an Agile project are conducted by a variety of team members in a cross-functional way. There is no 'they vs. us' but rather just one team.

4.3. Communication

Agile practitioners recognise swift communication as one of the main pillars of Agile project success (Eckstein, 2010). This recognition has fuelled the inclusion of communication into the set of core values laid forth in The Agile Manifesto (see Figure 1). There, its importance is underlined by defining 'individuals and interactions' as the topmost value.

As a consequence, most Agile methods require teams to be small and collocated. The size of the team needs to be restricted as the number of different communication channels increases dramatically with team size. For instance, a ten-member team has more than twice the communication channels of a seven-member team. Small teams enable everyone to know and communicate with every other team member when necessary.

Team collocation influences communication as well, as it is obviously much easier to discuss issues or answer questions if both parties sit in the same office or even at the same desk. Other important Agile concepts such as the daily Scrum meeting or XP pair programming rely on this being the case.

In an Offshore Agile project, this is the single most prominent Agile value that absolutely cannot be fulfilled. Having the complete offshore team onsite for any duration of time is not an option; neither is moving the onsite team to the offshore location. While this prevents Offshore Agile projects from adhering to all Agile values in full, it does not prevent the adoption of Agile and Offshoring. However, the limitations that are imposed by dispersed teams need to be understood, and measures need to be defined to overcome them where possible and to accommodate them where necessary.

The communication strategy and framework defines and orchestrates all measures as well as the tools necessary to implement them. This approach has a number of benefits, among them transparency concerning which communication channels to use to solve a specific problem, communicate a specific piece of information or get in touch with a specific person or role.

Tools	Examples	Voice	Video	Text	Document Sharing	Comm. recorded
Meeting Tools (dynamic communication)						
Voice calls	Telephone, VoIP, MS Communicator	✓	–	–	–	–
Video calls	Video Phones, MS Communicator	✓	✓	–	–	–
Screen sharing	MS Communicator	–	–	✓	✓	–
Instant messaging	MS Communicator (SMS via Communicator)	–	–	✓	–	–
Messaging Tools (delayed communication)						
Voicemail	Telephone, VoIP, MS Communicator	✓	–	–	–	–
Email	Outlook	–	–	✓	–	✓
Instant messaging	MS Communicator	–	–	✓	–	–
Message boards	SharePoint, Google Sites	–	–	✓	✓	✓

Figure 6: Sample of communication matrix mapping communication tools to communication needs

Figure 6 shows an example of how a communication strategy provides mappings of communication tools to specific needs. Here, each team member, onsite and offshore alike, can easily identify that in order to share a screen and collaborate on a document, MS Communicator screen sharing would be an appropriate tool.

In a similar vein, meeting schedules require more detailed planning, simply because onsite and offshore locations are usually not in the same time zone and, in fact, may be separated by a significant time difference. In such cases, it is important to identify corridors where all team members are in the office and are indeed available for a call. If this is not considered, meeting attendance will most likely be low, decreasing the vital flow of information. In extreme cases, it may become necessary for one side of the team to shift their work hours slightly; for instance, 8 a.m. US EST is 6:30 p.m. in India, which might require the Indian team to stay until the evening hours.

Enabling swift communication requires an investment in onsite / offshore visits, Agile and communication tools, and team building activities. Among the most effective methods used to bridge the communication gap are onsite and offshore visits by team members.

In both cases, staff visit the 'other' part of the team for a specific duration of time in order to work with them, but also to deepen the understanding of work environments, styles and ethics, cultural differences, building trust between individual team members and the two halves of the team as a whole. Team building events serve the same basic purpose in fostering understanding, communication, and trust within the Agile team. However, such events are often difficult to organise in a way that on- and offsite teams are able to attend.

Having team members travel between sites obviously requires a budget and will inevitably decrease the potential savings that are often the reason behind moving work offshore. However, money spent on this type of visit is money well spent. A number of industry leaders have demonstrated that the cost of having team members travel to collaborate is offset by the costs generated by misunderstandings or lack of trust.

Besides affording as many team members as possible the opportunity to visit the remote part of the team, it has shown that having at least one representative of the offshore team onsite is a vital benefit, since this team member can act as a communicator understanding both sides of the team. Depending on the size and complexity of the project, more than one such representative may be necessary. When considering the testing component of Agile projects specifically, this role becomes a dual one: the onsite representative not only represents the offshore team onsite but also represents the test-focused team members when communicating with architects and developers.

Lastly, adhering to established and well-understood communication paths helps all team members understand the status of the affected deliverables, which process steps have been taken care of and which steps are still open. This needs to be documented precisely enough for all parties to understand the process and the tools involved.

4.4. Governance

Managing a dispersed team efficiently and effectively is always a challenge; managing a dispersed Agile team even more so. A number of best practices have emerged that support successful governance even in such a challenging context:

- Choose appropriate and helpful dashboards
- Track status using metrics
- Establish a central collaboration hub
- Select and adhere to Agile processes and practices
- Clearly define exceptions where Agile practices cannot be used

Starting with a dashboard comes naturally in an Agile context. Here, all experienced team members are expecting full transparency concerning the status of individual tasks for the current iteration, as well as transparency about how the team as a whole – and to a certain extent the overall project – is doing.

Usually, a blank wall in the team's office is enough to serve as such a dashboard, as any team member can stroll by and have a look on the way to the coffee machine. Figure 7 illustrates what such walls often look like with the individual story cards, the associated tasks and their status clearly visible, as well as aggregated information such as sprint goals and team performance on the right.

With dispersed teams, this obviously is not a viable approach. However, the Agile wall serves an important purpose for the Agile team and cannot simply

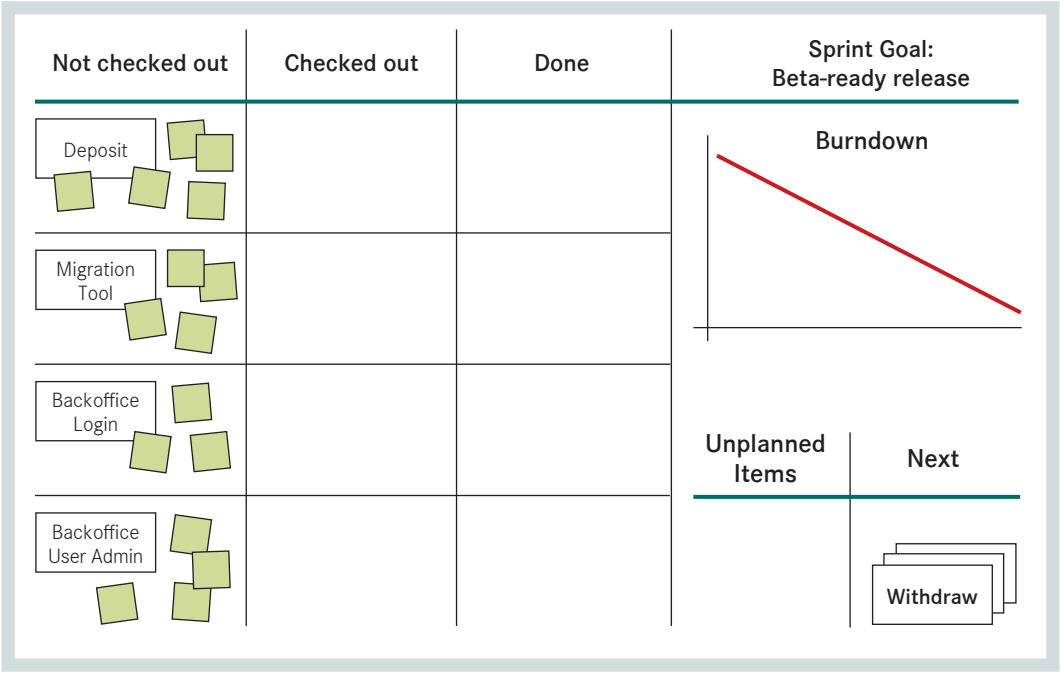

Figure 7: Illustration of an Agile wall (DeFauw, 2011)

be omitted without compromising the Agile project management approach. Therefore, adequate project management tools and a virtual Agile wall need to be set up so that the Agile team has the necessary information to conduct the crucial self-management. For instance, the testers in an Agile team can plan their testing activities based on the completion of user stories on the wall. If a story is 80 % complete, then the Agile tester knows that it will be available for testing shortly, for instance within the following business day.

In this context, metrics serve the purpose of increasing transparency and providing additional visibility into the status of the project, both for the current iteration (e.g. the current sprint in Scrum) or for the project as a whole. A wide variety of metrics is available and has to be tailored to the project's needs in order to provide a concise picture of the current state.

What is true for Agile dashboards is also true for the remaining tool chain. Agile puts a strong emphasis on tools in order to enable collaboration wherever possible, and in order to automate tedious chores that eat up valuable time that could better be spent on productive work. Subsequently, it is of paramount importance for Offshore Agile projects to establish a single location that can be used by all team members to share information. In practice, a wide variety of specific tools are being used to accomplish this goal, ranging from Wikis to full-blown collaboration solutions such as Microsoft SharePoint.

Any of the available technologies can be made to work as a collaboration hub. What is more important is the concept of what to share on such a hub in which way. Information to be shared includes the following:

- Vision, goal, or big picture of the initiative, programme or project
- All relevant documents in their relevant version
- Product information
- Centralised risk logs
- Release notes
- Environment details
- Discussion forums
- Project calendar, leave calendar, events
- People / team news

Tools aside, two aspects of governance increase in importance when an Agile approach is used in an Offshoring context: which Agile practices to adhere to and which to discard. Those practices that are to be kept need to be made crystal clear to all participants and need to be trained (see Section 4.2) and communicated (see Section 4.3) to all team members. On the other hand, those practices that cannot be adhered to need to be made explicit as well.

For instance, keeping a daily Scrum meeting – even with dispersed teams – is a prerequisite for success. Therefore, it needs to be clear that each site is required to participate in the meeting. Ideally, this involves all team members from all sites so that everyone will be able to communicate issues or voice concerns. In some cases, it may suffice that each site is represented.

Agile practices such as having a collocated team cannot be upheld with Offshoring. As a consequence, the balance of implicit vs. explicit documentation changes since more information has to be made explicit by keeping it in reference files. Where usually informal communication between team members in the same office would have been adequate to resolve issues, a dispersed team requires more explicit documentation. The appropriate amount and level of documentation needs to be specified, however, so that all team members know how much documentation work is expected of them.

Unlike orthodox Agile teams, where a strong emphasis is laid on the self-governing forces within the team, Offshoring requires more explicit mechanisms of conflict resolution. This is due to the fact that it is more difficult to resolve issues between dispersed teams, but also because Offshoring often means that not all team members are part of the same organisation and, thus, conflicts of interest may arise which require escalation to be resolved. Therefore, escalation paths and mechanisms need to be clearly determined at both project and (if applicable) programme level.

In addition to the roles and responsibilities defined by the Agile approach (e.g. the Scrum Framework with Scrum Master, chickens and pigs), Offshore Agile requires some further roles to be in place beyond that. For instance, the offshore part of the team benefits from the presence of a local team lead or even outright Scrum Master who can act as a point of contact resolving misunderstandings and open issues.

4.5. Automation

Automation of repetitive tasks is as much of a cornerstone of Agile practices as is the iterative, communicative nature of teamwork. The rationale behind this is obvious: by automating tedious, time-consuming work that can be done by a tool, people gain time to concern themselves with work that no tool could do for them, which increases team productivity.

In true Agile projects, this credo has been perfected using tools such as continuous integration and daily /nightly automated tests in order to increase efficiency as much as possible. At the same time, an entire tool ecosystem has sprung up to support these approaches. As many of those tools originate in a grassroots or community effort to implement tools specifically for the purpose of assisting Agile software development, many tools are free or Open Source software. This results in an environment with a multitude of small specialist tools (such as unit test, test coverage or acceptance test tools) that are well integrated with each other and with the integration and dashboard solutions.

This context is quite different from that of traditional software development projects, where development and testing are usually dominated by heavyweight commercial tool chains such as Microsoft Team Foundation Server, HP ALM, or QuickTest Professional.

For an Offshore Agile test team, it is crucial to be able to adapt to this different environment in order to add value to the team as a whole. In the Agile space, the entire process of software creation and testing is eligible for automation. This includes the following:

- Test data creation
- Unit / component testing
- Non-UI testing, i.e. API, web services
- GUI testing
- Non-functional testing (e.g. performance, security, usability)
- Continuous integration of build and deployment

When planning test automation activities in the context of an Offshore Agile software development project, the question arises of how to plan and conduct test automation, which in itself is also a software development project, albeit on a much smaller scale. Moreover, test automation activities will most likely be conducted by the offshore segment of the team.

Naturally, the best results have been achieved when test automation activities themselves have made use of the same approach as the overall software development project, meaning that for test automation the same approach and same principles should be followed. In a best-case scenario, the test automation sub-project can be integrated seamlessly into the overall project management and reporting infrastructure.

This approach guarantees a number of benefits, i.e. the following:

- It prevents test automation from taking a backseat when the release is near
- There is a clear division of labour
- The QA team working on the release can focus on the new features
- The automation team can focus on the regression of a much larger number of old features (as regression is only performed for the stable parts, anyway)
- The progress and success of the automation is clearly visible

5. Case Study

The best practices presented in Section 4 are derived from a number of Offshore Agile projects delivered to customers in Europe and North America. The Offshore Agile test effort for Copyright Clearance Center (CCC) Inc., based in Boston, MA, showcases very well the success of IT projects that make intelligent use of Offshoring for testing work packages.

CCC is a global rights broker based in the US, for the world's most sought-after materials, including in- and out-of-print books, journals, newspapers, magazines, films, television shows, images, blogs, and e-books.

At this customer, as a QA & Testing arm, SQS has been responsible for the testing of numerous initiatives and projects which are conducted simultaneously across multiple applications. This has included functional testing as well as localisation, internationalisation testing, and non-functional testing, particularly performance and security testing. Some of these processes have been automated by way of GUI, services and API test automation. The delivery of many initiatives is based on the Scrum Agile software development approach.

With this customer, SQS has found that the success factors specified in the present paper were key to providing a value to the customer by successful delivery of QA & Testing services.

The specific training approach chosen was to conduct a pre-kick-off Agile induction boot camp for all team members, covering Agile concepts as well as customer-specific processes. Additionally, recorded training sessions from existing team members were used to up-skill newbies quickly and efficiently. On top of that, the resources of an SQS-wide virtual Agile community – the 'Innovation Group Agile Testing' – were tapped and assisted staff in answering Agile-specific questions.

From a communication point of view, the biggest challenge was to keep the offshore portion of the team updated in a continuously changing environment. The solution was to be represented at the daily Scrum meetings by onsite and offshore SQS team members. The regular onsite visits helped tremendously to foster trust and understanding. Technically, email distribution lists and an online collaboration platform were used to communicate information such as the progress and updates on stories, or information about defects fixed in the upcoming build.

6. Conclusion and Outlook

The objective of this whitepaper is to describe best practices to help meet the challenges of implementing both industrialisation (Offshoring) and craftsmanship (Agile) at the same time. Whereas with tangible goods such an attempt would be downright impossible, with software it is a workable approach. However, many challenges need to be overcome for Offshore Agile to be successful. Most importantly, orthodox positions need to be revisited, and all parts of an approach that do not help or are not applicable need to be thrown overboard. This results in an approach that is neither pure Agile nor pure industrialised software development, but that is suitable for the specific context in which it has been developed.

This paper has not touched on a number of issues, either because those issues are quite specific and not relevant to a broad audience or because they do not pose a particular challenge to Offshore Agile Testing. And although there are many challenges that are inherent in either Offshoring or Agile software development methods, they are – as such – not covered by the scope of this whitepaper.

Nevertheless, questions such as how to negotiate the role of testers in an Agile team, how dedicated testers can improve the quality without interfering with Agile values, or specific technical or compliance challenges need to be understood, analysed, and solved in order for Offshore Agile Testing to be successful.

What has become clear, though, is that there is no principal obstacle that would prohibit the successful implementation of Offshore Agile Testing. The best practices presented above constitute the first steps towards empowering successful and fun Agile teams, regardless of their geographical location.

7. Bibliographical References

Agile for All. Introduction to Agile. [Online] 2008. [Cited: 2012.]
http://www.agileforall.com/intro-to-agile/

Computer Economics. IT Outsourcing Statistics 2010 / 2011: Outsourcing and Offshoring Trends, Cost / Service Level Experiences, and Analysis for 11 Outsourced IT Functions. Irvine, CA: Computer Economics, Inc., 2010.

DeFauw, R. Agile in Action. [Online] 10/08/2011. [Cited: 2012.]
http://www.perforce.com/blog/110810/agile-action

Eckstein, J. Agile Software Development with Distributed Teams: Staying Agile in a Global World. New York: Dorset House Publishing Company Inc., 2010.

Krill, P. Agile Software Development Is Now Mainstream. [Online] 22/01/2010. [Cited: 2012.]
http://www.cio.com/article/522165/Agile_Software_Development_is_Now_Mainstream

Leclerque, K. Outsourcing im Jahr 2020. [Online] 29/09/2010. [Cited: 19/07/2011.]
http://www.cio.de/knowledgecenter/outsourcing/2247905/index.html

Simon, D., and Simon, F. IT Industrialisation as Enabler of Global Delivery. 2012 16th European Conference on Software Maintenance and Reengineering. Szeged: Institute of Electrical and Electronics Engineers, 2012.

The Standish Group: Chaos Report 2009. The Standish Group International, Inc., 2009.

Tyler, R. SCRUM (Agile Methodology). [Online] 19/08/2010. Confessions of a Systems Developer. University of Glamorgan Blog: Learning and Corporate Development. [Cited: 2012.]
http://lcd.blogs.glam.ac.uk/2010/08/19/scrum-agile-methodology/