Introduction

With the growing adoption of mobile payments technology, competition is under way amongst technology providers to become the supplier of a worldwide mobile payments standard. The current lack of standardisation has created a number of technical paths for mobile payments and has also highlighted the lack of interoperability. Organisations planning to provide customers with mobile payment capabilities need to select one of the available technologies in the hope that it will become the dominant standard, or they risk being left behind.

So how can organisations ensure that they do not find themselves backing a losing horse? One option is to accept that change is inevitable and retain the flexibility to change horses mid race to achieve success. Once the need for flexibility is established, it becomes clear that a dynamic approach to software development is key to the successful adoption of mobile payments. This paper seeks to establish how Agile development methodologies provide this flexible approach, which embrace practices that provide organisations with the confidence to make and release changes rapidly without impacting quality.

Authors: Jerry Stubbs (Associate Director, Agile)
  jerry.stubbs@sqs.com
  Paul Wilford (Principal Consultant)
  paul.wilford@sqs.com

SQS Group Limited, United Kingdom

Published: March 2014
1. The reason why mobile payments needs Agile

1.1. The mobile payments space

Khalid Fellahi¹ from Western Union notes that: “A year ago digital payments cross-border accounted for 1% of the company’s business. The digital push has seen that grow to 35% in a year, driven mainly by new customers,” and that “most of these customers don’t want to go to a physical location to transfer money; they want to do it electronically”.

Cross-border mobile payments are a major challenge; as Karl Rieder² of GFT Group observes, “much of the work being done in mobile payments, be it by banks, operators or third parties, is country specific right now.” This phenomenon increases the complexity of moving services across borders. Germany, for instance, doesn’t have the same m-payments tools as Spain. Even within some banks, tools for handling mobile payments are incompatible between different countries in the same bank brand. This is why companies such as PayPal, which have no international issues, are so compelling.”

1.2. The mobile payment market

As with any new technology, early attempts to generate income from the mobile payments space may involve high risk strategies. This is largely due to the lack of successful approaches from both a technology and, perhaps more importantly, customer behaviour point of view.

Take Near Field Communication (NFC) payment technology; according to Gartner, uptake has been significantly slower than expected. In fact, Gartner has reduced its forecast transaction value for NFC by more than 40 per cent. This has led some analysts to assert that ventures such as Google Wallet and Isis have largely been a waste of investment³ ($300m in the case of Google Wallet).

Agile methods can help organisations to protect themselves against such situations by supporting small lightweight releases that are ideal for testing a business case or theory with real users. Feedback from this development approach can be invaluable in decision making and changes in priority can be quickly integrated in the development process.

Quality is essential in these situations. Jerry Weinberg⁴ stated that: “Quality is value to some person” (and James Bach’s⁵ wry suffix “who matters”). It follows, therefore, that a sound business theory can all too easily become a failure due to poor execution. This is where the value of testing becomes key, with a strong and easily maintainable test automation framework. High availability of quality information, from daily automated tests, will provide business stakeholders with the confidence to release products to the market regularly. It also arms development teams with the necessary confidence to make changes, even where the risk is perceived to be high.

In 2012, when Deloitte asked of a Chinese audience: “According to your knowledge, which [solution maturity] phase do the following mobile payment technologies belong to [in China]?” Its respondents considered SMS to be the most mature (see Figure 1, next page) and NFC-based payment, the least, with 50 per cent believing that it is still in a trial phase.
2. Adapting for successful mobile payments strategy

Whilst some successful programmes have been delivered through the Waterfall methodology, this approach is best suited to predictive projects. If all of the work can be defined up-front and little is likely to change, this step-wise and plan-driven process has its place.

However, if projects are more innovative and contain elements of team exploration to satisfy customer requirements, specifications are likely to change frequently, and experience has shown that a more iterative approach to development has a greater likelihood of success.

Developing software for the mobile market is a demanding task and vendors need to adapt quickly to fast changing platform designs and customer expectations. Therefore, an alternative to a Waterfall approach may be more appropriate to decrease development costs, increase deployment frequency and to continually drive-up quality.

As Robert Holler, CEO of VersionOne states, “companies such as Sabre, Sprint, Nortel, Symantec, Fidelity, Borland, Qwest, and hundreds of other leading technology and software organisations have begun employing agile development methods with considerable success” – and commercial advantage.

As Magnus Jern says, “in mobile development, agility is especially important as it’s difficult to anticipate how a service will work across various mobile devices until you’ve tried it out”. So, what is agility and how can it benefit mobile payments development?
3. Agile methodology

The Agile Development Methodology is actually a collection of iterative and incremental methods that have evolved since IBM’s Los Angeles office began experimentation in early 1957. In the mid-1990s, the threads of several similar approaches were beginning to weave together: RUP\(^9\), Scrum\(^10\), Crystal Clear\(^11\), XP\(^12\), FDD\(^13\) and DSDM\(^14\) (which was introduced to add discipline to RAD\(^15\)) and ASD\(^16\) to form what we now know as Agile.

Agile itself, though, was born in Snowbird, Utah, in February 2001, when 17 developers\(^17\) met to discuss lightweight development methodologies. Many of these developers had been instrumental in the emerging threads listed above. Between them, they created the ‘Manifesto for Agile Software Development’ (see Appendix 5.1). The team defined 12 principles upon which true Agile should be based (see Appendix 5.2).

In practical terms, Figure 2, below, shows the cycle of events that is typical of most Agile approaches. It serves to show that the approach is:

- iterative
- disciplined and well-structured
- made up of defined roles
- highly collaborative
- driven by a need to add customer value

![Figure 2: Agile Activity Flow](image-url)
In relation to mobile payment application development:

- Customer value is led by ensuring that payments are successful, secure and fast.

- Key to Agile’s success is team collaboration – a laser-like focus on delivering value and a shared responsibility of code quality.

- Several techniques have evolved in support of these objectives, three of which are described below – all will deliver benefit if deployed for the development of mobile payment apps.

### 3.1. Test Driven Development

Test Driven Development (TDD) is not a new concept: the process of driving development by first writing Unit Tests has been considered good practice since Kent Beck developed the technique in 2003\(^{18}\). Agile principles take TDD beyond Unit Testing and use Acceptance Tests to define what should be developed, while creating tests that can be understood by business representatives. There are many advantages to this approach, and SQS has helped clients to implement these practices on a number of occasions with benefits including:

- living documentation of the system, the tests also describe how the system should behave

- ability to make changes quickly with comprehensive automated regression suites available for execution in short timescales

- confidence to make changes, knowing that developers can quickly verify that a system has not been, or more importantly will not be, damaged by a ‘risky’ change

- shifting focus to defect prevention by instilling a culture of building the right code the first time.

This Agile approach enables teams to significantly increase test coverage and visibility, as well as build quality early in the process. Automated tests also enable repeatability, which is a powerful ally in the competitive and varied mobile payments market, with the ability to execute the same tests on many platforms.

### 3.2. Specification by Example

The practice of Specification by Example builds upon the idea of defining Acceptance Tests to drive development\(^{19}\). In this approach, a description of how the system works is accompanied by examples which elaborate and define what should happen in different scenarios.

In, Gojko Adzic’s book\(^{20}\), he asserts that the whole team should work together on defining these examples, using their combined knowledge and imagination to fully understand what is required. When the examples are defined in a table, they can be mapped directly to the business logic in the code and executed against that code using tools such as Cucumber JVM or FitNesse. This provides a powerful way to define features in collaboration with customers; demonstrate that the software does what it should; and also document how the system should work.

In the mobile development world, this approach enables developers to document how the system has been created for a wide variety of platforms. Tests can be copied and modified to build device specific suites, which provide visibility to support rapid modifications in the future.
3.3. Exploratory Testing

There are some elements of quality that are difficult to verify through automation. In these cases, there is often value in user testing as testers can spot unforeseen behaviour, comment on visuals and usability and provide an informed opinion on the experience of using a product.

Exploratory Testing can provide significant value in this area. However, it is worth noting that structure, discipline and reporting processes should be implemented to make the process efficient and realise the greatest benefit.

Experienced testers can carry out structured exploratory testing sessions with specific objectives and feedback tailored to meet the needs of the situation. With a prioritised list of objectives, sessions can be tracked to report on progress and stakeholders kept informed of coverage and results. As testers apply their knowledge and experience to each session, they can follow unique paths, and explore the system in a more intelligent way.

Exploratory testing is a powerful technique and is best performed by experienced testers freed from repetitive manual testing by embracing the automated testing so essential to Agile’s success.

4. Conclusion

In a chaotic and quickly evolving mobile payments market space, requirements for customer applications are changing minute-by-minute. This is driven by technology advances, end user expectation and by vendors seeking to create USPs. Only as the market starts to mature, will it be possible to establish where the highest value propositions are to be found. The pioneers in this field must build solutions in a way that enables them to gain rapid feedback from the market and to respond quickly when trends start to appear.

Without a development methodology designed to embrace or even welcome these evolving trends, there is a very real risk that the wrong product will be delivered late and to a market that is no longer there.

Organisations that have embraced Agile wholeheartedly, and this usually includes a significant culture change programme, are better able to deliver the right product at the right time. Experience shows us that the key goals for these organisations should be: executable specifications; flexible automation frameworks providing the most appropriate coverage at the most appropriate levels; and fully integrated development and test processes with continuous integration, delivery and defect prevention concepts at their core.
5. Appendix

5.1. The Agile Manifesto

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

<table>
<thead>
<tr>
<th>Individuals and interactions</th>
<th>Process and tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working software</td>
<td>Comprehensive documentation</td>
</tr>
<tr>
<td>Customer collaboration</td>
<td>Contract negotiation</td>
</tr>
<tr>
<td>Responding to change</td>
<td>Following a plan</td>
</tr>
</tbody>
</table>

That is, while there is value in the items on the right, we value the items on the left more.”

5.2. The 12 principles of Agile

The 12 principles upon which Agile is based are:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software

2. Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale

4. Business people and developers must work together daily throughout the project

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done

6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation

7. Working software is the primary measure of progress

8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely

9. Continuous attention to technical excellence and good design enhances agility

10. Simplicity – the art of maximizing the amount of work not done – is essential

11. The best architectures, requirements, and designs emerge from self-organizing teams

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly
5.3. References

1) Khalid Fellahi, senior vice president and general manager of Western Union Digital Ventures
2) Karl Rieder, delivery manager at financial technology provider GFT Group
4) http://www.geraldmweinberg.com/Site/Home.html
5) http://www.satisifice.com/aboutjames.shtml
7) “Mobile Application Development; A Natural Fit with Agile Methodologies”, Robert Holler, President and CEO of VersionOne, 2006
8) “Agile Predictability in Mobile”, Magnus Jern, November 5, 2012
9) IBM’s Rational Unified Process, published in 1999 by Ivar Jacobson, Grady Booch and James Rumbaugh. RUP’s lifecycle is divided into four phases: Inception; Elaboration; Construction and Transition
10) In their book, “New New Product Development Game”, Hirotaka Takeuchi and Ikujiro Nonaka describe Scrum as “a flexible, holistic product development strategy where a development team works as a unit to reach a common goal”. They liken this to rugby.
11) Alistair Cockburn described a family of methodologies under the heading of Crystal, which focus on “efficiency and habitability as components of project safety”. He states there are three fundamental properties to CC: Frequent delivery of usable code; Reflective improvement; Osmotic communication – preferably by being co-located
12) Kent Beck described XP in his book “Extreme Programming Explained”, published in 1999. Amongst other ideas, Beck was first to propose the “practice of test-first development”
13) Feature Driven Development’s principal tenet is to “…deliver tangible, working software repeatedly in a timely manner”
14) Dynamic Systems Development Method fixes cost, quality and time at the beginning of the project and uses MoSCoW to prioritise scope. An essential feature is continuous user/customer involvement
15) Rapid Application Development uses minimal planning in favour of rapid, throw-away prototyping
16) Adaptive Software Development grew from RAD. It replaces Waterfall with a repeating series of speculate, collaborate, and learn cycles
17) Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Stephen J. Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas
18) Kent Beck is credited with having developed or ‘rediscovered’ the technique, stated in 2003 that TDD “encourages simple designs and inspires confidence”
19) Specification by example is also known as Example-driven development, Executable requirements, Acceptance Test-Driven Development (ATDD) and Agile Acceptance Testing
21) Cem Kaner, who coined the term in 1983, now defines exploratory testing as “a style of software testing that emphasises the personal freedom and responsibility of the individual tester to continually optimize the quality of his/her work by treating test-related learning, test design, test execution, and test result interpretation as mutually supportive activities that run in parallel throughout the project.”
22) The left-side items are taken to mean: Individuals and interactions – in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming; Working software – working software will be more useful and welcome than just presenting documents to clients in meetings; Customer collaboration – requirements cannot be fully collected at the beginning of the software development cycle, therefore continuous customer or stakeholder involvement is very important; Responding to change – agile development is focused on quick responses to change and continuous development

23) http://agilemanifesto.org

24) http://agilemanifesto.org/principles.html