Outsourcing our Software Development with Agility

Zhu Qingyu (Student ID: 3035419080)

November 17, 2017

Abstract

We put forward an approach to outsource the bank software development based on agile manner. This approach focusing on three aspects, which are: manage project scope, manage project participants and manage project process, aims to mitigate the important risks CIO would face when outsourcing software development. First, we brief introduce the agile approach to software development and its benefits and challenges. Second, from the viewpoint of CIO, we present the objectives and important risks of software development outsourcing. Third, we put forward the approach mentioned above with a summary.

Contents

1 Briefly describe the Agile approach to SW development and its benefits and challenges.

1.1 Introduction of agile approach to SW development

In brief, agile approach is a way of executing software development management. So it is a project management methodology. To describe the agile approach to SW development, we will brief answer three questions: Why does it exit? How did it get here? And what it is?

1.1.1 Why does it exit?

Before people start doing agile software development, there was a development process called waterfall. In waterfall development process, people go through a number of mostly sequential phases beginning from gathering requirements to designing and then moving into implementing then to testing and then finally launching the project and maintenance it. Each phase is completed before the next one starts. However, it's really hard to finish these things within the planned time since there are many changes and we don't know what will happen at the end.

1.1.2 How did it get here?

In February 2001, the *Agile Manifesto* was put forward. It has four core pillars: 1) individuals and interactions are valued over processes and tools; 2) working software is valued over comprehensive documentation; 3) customer collaboration is valued over contact negotiation; 4) finally and perhaps most importantly, responding to change is valued over following a plan. In summary, the *Agile Manifesto* figures out that, change should be embraced since the unpredictability during the software development.

1.1.3 What it is?

Although there are many implementations of agile approach to software development, the most accepted steps in an agile software development approach can be describing as follows:

1) The first thing is to identify the type of project. In agile approach, understanding customers' or users' visions and requirements before starting development is very important. Agile software development usually begin with a collection of investigations to identify customers' or users' business requirements, goals and business environment. Every relevant person including the customers/users, CIO, project manager, designers, and developers should be involved in this steps to ensure a clear understanding of the whole project.

2) And the second one is to create a preliminary product backlog. During the first step, the team works together to create a high-level product backlog, a overview of the project plan and list of all the features that would be useful to the customers and users. The product owner works with the customers to prioritize these features. By allowing the customers to determine priority, the team focus on implementing the highest value features before moving on to lower value features.

3) The third steps is to plan sprints. After the team has clearly identified the type of project and has created a preliminary product backlog, the team delivers product features through a series of time-boxed sprints. According to the size and scope of the project, each sprint would be last one to four weeks. During each sprint, a more detailed and smaller product backlog can be implemented and delivered.

4) The last task is to continue the cycle. Additional sprints are operated as needed to deliver additional features and incorporate feedback in previous sprints, reviews, and testing. Each successive sprint is both iterative and incremental. In these cycles, develop team would provide improvements to work completed in previous sprints and add new features to the system.

1.2 Benefits

1.2.1 From the viewpoint of customers & business stakeholders

1) More satisfaction. In the agile software development, customers can more constantly participant in the development process and sooner get solutions to the problems they value most. So customers have chance to clearly express their requirements to the development teams. This reduces the risk that the product not meeting the needs and requirements of the customers. With everyone coordinating together, the project leads to a fruitful completion.

2) More flexible. Although the agile teams are focused to deliver timely solutions that meet business stakeholders satisfaction, the business stakeholders still can re-prioritize and refine any product backlog. Changed or new backlog items can be scheduled for next iteration, therefore providing the opportunity to implement changes in just a few weeks. Additional, using agile approach, business stakeholders have frequent and early opportunities to see and have a say in what is being developed.

3) More cost-effective. Since the agile project is time bound, it is also bound by a fixed budget rather than high up-front costs in investigation, documentation, and contracts. The user requirements and scope not only keep changing but also have to be managed within the specified budget. So agile approach can ensure that the project is delivered on time as well as cost effectively.

4) Improved ROI. Firstly, in agile approach, business stakeholders not being forced to rely on features that proposed before but can adjust and deploy more quickly based on today's need. So this can keep them ahead of the competition in fast-moving markets. Besides, agile approach means fast product releases and can quickly produce a working version of software. This early deployment of functional parts help business stakeholders begin to see real ROI more quickly.

1.2.2 From the viewpoint of agile teams

1) More efficiency. Firstly, the agile teams are self-organizing teams. individuals are highly motivated to bring out better outcome from the development projects. The agile approach guides developer teams to execute sprint planning within the required time period. Secondly, discussing issues in daily scrums can help to promote development efficiency. Furthermore, changed and new features are delivered frequently and quickly by using the time box technology. This further provides chances to release the software project earlier than its designated date.

2) Higher quality. Firstly, agile teams identify and investigate user requirements just in time so that the understanding of the product features is as relevant as possible. Secondly, splitting the project into manageable small units, the development teams can concentrate on producing and testing high-quality code. Thirdly, the regular testing and reviews after each iteration help to improve the project quality as any mismatches are identified early and defects are fixed quickly.

3) More confident. A agile team may have a list of the most important things they need to work. When they finish the most important thing on that list, they move to the next most important thing. During this process, developers can feel valued because they're working on things that actually matter and will receive frequent in-depth and valuable feedback from business stakeholders or customers. Besides, with a good understanding of user requirements, developers are more likely to use the best techniques for keeping the design under control without excessive work. A testable and reliable result can make developers feel more confident.

4) More solidarity. Agile team focus very heavily on the collaborations between people like business stakeholders, customers and teammates instead of process and tools. Interaction is very important because it is the input from developer teams, customers and business stakeholders. It will ultimately makes software project a success as opposed to what tools they use. Continual collaboration throughout the entire development cycle enables everyone involved to build up a good working relationship that will be based on trust. This trust based working relationship is crucial when building software incrementally. When developers are friends, or at least more familiar with each other, it makes working together much easier.

1.3 Challenges

1.3.1 From the viewpoint of customers & business stakeholders

1) Difficult to get involved The agile approach states that customers should be part of the development process from analysis and design to implementation and maintenance. However this principle is very demanding on the customers or business stakeholders representative's time and requiring a big commitment for the duration of the project. Some customers or business stakeholders may be not involved in the decision making process until the project is all done. Usually, They are busy and they have other things to do than to talk to programmers all day.

2) Unclear customer requirements Most of the time, customers or business stakeholders do not know what they really want in their future system and it becomes a roadblock for them to get involved in the project development process. Customers and business stakeholders think they have a clear idea but they do not. For example, business stakeholders tell developers that they want to track people's credit. To business stakeholders, that's clear and precise. But to developers, they need to know who the people are, what the credits are, when they expire and what rewards earned for many credits and so on. So how to put forward clear and timely requirements is a big challenge to customers and business stakeholders.

1.3.2 From the viewpoint of agile teams

1) Harder for new starters. Firstly, agile requirements are normally barely sufficient. Requirements are clarified just in time for development and can be documented in much less detail due to the timeliness of conversations. This means less information available to new starters in the team about features and how they should work. Besides, It can also create potential misunderstandings if the teamwork and communication are not at their best, and difficulties for team members like testers that are used to everything being defined up front.

2) Increasing cost relating to testers. Testing is integrated throughout the lifecycle. It means that testers are needed throughout the project and this significantly increases the cost of resources on the project. Although frequently testing does have the effect of reducing some very significant risks which can cause many projects to fail, the cost of a long and unpredictable test phase can usually cause huge unexpected costs when a project over-runs.

3) More pressure. Firstly, agile teams need to complete each feature 100% within each sprint or iteration. This can make developers feel more pressure and mentally quite tiring. Secondly, if customers cannot give specification documents, agile teams basically have a long meeting to make a specification.It may be hard when they started it. Thirdly, most agile approach remove the cubicles and set up collocated team space because cubicles promote isolation and the agile approach relies heavily on face-to-face communication and network. However, some developers may feel intense and be less productive in a open space.

4) Less predictability. Since user requirements emerge and evolve throughout the development, agile approach is very flexibility. However, there are two big flip sides to this principle. One is the potential for scope creep, which can increase the risk of ever-lasting projects. The other is that there is much less predictability, at the start of the project and during, about what the project is actually going to deliver. This makes it harder to identify a business case for the project, and harder to negotiate fixed price projects. Without the maturity of a clear vision, and the discipline of fixing time-scales and trading scope, this is potentially very risky.

2 In the specific context of SW Development, briefly describe the objectives of outsourcing, as well as important risks that you would expect to face as the CIO.

2.1 The objectives of outsourcing

2.1.1 Time objectives

1) Working at 24/7. Outsourcing software development in other countries can take advantage of the time zone differential, allowing almost 24/7 operations. When talented professionals are working on the project around the clock, the software development seems to take less time. By this way, companies can get their products to markets more quickly than their competitors can.

2) Time spent wisely. On the one hand, managing an in-house development project can take valuable time away from companies' core business duties. But outsourcing frees companies to devote more of their attention to other matters like their main tasks and duties. On the other hand, the number of developers needed can change over the process of the project. At times, more developers are required to meet goals, but at other times, the project requires fewer. Outsourcing can flexible reassign employees according to the needs of each project at different time.

3) Time savings. And most importantly, outsourcing software development can save much time. Firstly, recruiting, onboarding and training new employees can be time-consuming and costly. But when companies outsource, they do not have to spend time recruiting, hiring, training and housing employees for short-term projects. Secondly, it is very common for project deployment times to be substantially shorter when outsourcing software development than when developing a similar project in-house. Choosing to develop a project using agile approach when outsourcing, individual modules can often be deployed separately and in a matter of weeks. This allows companies to begin realizing a return on their investment quickly. As the saying goes, "time is money."

2.1.2 Finance objectives

1) Less labor cost. Firstly, outsourced labor in developing countries may cost less than the same labor performed in-house in Western Europe or North America, particularly for low-level tasks. Additionally, outsourcing allows CIO and the bank to utilize contracted employees rather than the ones that are full-time. This means they don't have to pay a salary and just have to pay contracted employees for the hours they actually work. Many prefer to use a time tracking software that shows CIO and the bank exactly how many hours the remote workers are working, what projects they are spending time on, and when they sign in and sign out during the day.

2) Less operation cost. Outsourcing can also save some "soft" operation cost. Firstly, when hire a remote worker, companies don't have to pay costs of relocation, don't have to offer benefits, don't have to have a more expensive insurance policy, don't have to pay for a bigger office space and don't have to up front more money to train new employees. Besides, companies don't have to buy expensive software and hardware because outsourcing is resource sharing.

3) More capital funds. Since outsourcing reduces the need to invest capital in noncore business functions, therefore bank can make capital funds more available for core areas. Outsourcing also can improve corporate financial measurements by eliminating the need to show Return on Equity(ROE) from capital investments in noncore areas.

2.1.3 Resource sharing objectives

1) Sharing technological skills. Outsourcing allows companies have immediate access to some of the best and brightest information-technology professionals by going overseas and bypassing the gaps in hiring pools in more developed countries. If stakeholders need some software particularly coded or they don't understand the technology, outsourcing software services can provide them with all these skills. It is a good way for the bank to improve the technical competence.

2) Access to new market area. Some companies think about outsourcing part of SW development because they do not have access to the required resources within their own country. For example, if a bank would like to expand its operations to a new geographic area, outsourcing is a viable and important alternative to building the needed capability from the ground up.

3) Free resource for other purpose. Every company has limits on the resources available to it. Outsourcing permits a company to redirect its resources from noncore activities to activities that have a greater return in serving the customer.

2.1.4 Management objects

1) Simpler onboarding management. It is much easier to finding the right remote worker to outsource a task than to finding a new employee to bring into department office. When departments hire a remote worker, they won't have to worry about managing the training and benefits of employees. This also includes the fact that they don't have to account for sick leave, offer in-house performance reviews, or give raises based on an in-house contract. With a remote employee, departments are on a simple contract agreement with an external employee, and that's all there is to it. Remote workers really simplify the management process.

2) Simpler project management. The Internet, social media and a variety of online collaboration tools have certainly allowed CIOs to tap into the progress and status of their outsourced projects. When outsourcing software project, CIOs can just post a project on the fly and fill the position in a timely manner. Likewise, project cancellations, deliverations, and payment schedules can easily be managed. Outsourcing software development would create an automated compliance system that will reduce human follow-ups in business processes.

2.2 The important risks of outsourcing

2.2.1 Time risks

1)Delayed response. When outsourcing software development to other countries, the time difference between outsourced vendor and the bank can be about more than half day difference and it makes communication hard, because a question can be solved only the next day. For example, if the vendor and the bank are working in the similar time zone the CIO of bank can check in easily to see if outsourced partners are on track. If they work a significantly different time zone then the CIO of bank can't check in, so if they are heading down the wrong path, CIO won't know until the next day. In this case, they have just lost another day.

2) Personal time is occupied. Besides, the time different can also effect CIO's personal time even his/her family time, because of conference calls in late night, early morning and weekend. Look at this brief quote: " For the past 30 months, my day (Sunday through Thursday) has ended spending from 30 minutes to two hours on Skype orchestrating progress for FullContact with guys in a timezone an awkward 12.5 hours ahead of mine. I've conservatively spent 750 hours of my life (early morning or late night) on Skype with India in the past two and-a-half years, with the only real break being holidays or vacation." [1]

2.2.2 Finance risks

1) Unclear implicit costs. Cost savings is one of the most important reasons for CIOs and banks to outsource part of software development. Costs include explicit costs and implicit costs: the former including labor costs and "soft" operation costs can be more clearly observed. However, the implicit costs include the cost of the exchange and communication of outsourcing project management, the variable cost dealing with outsourcing content, and these costs may bring difficult estimation of the specific workloads, resulting in actual cost much higher than originally estimated cost. Although the whole projects achieves the expected cost savings, short-term cost savings may have potential risks and will constitute an roadblock to the development of the bank.

2) Fluctuating economics. Because the updating speed for software technologies and products is very fast, the underlying economics in IT industry usually shift very fast. For example, a mainframe system that cost \$1 million in 1965 costs less than \$30,000 today and probably will cost 20% to 30% less next year. This makes it difficult for CIOs to evaluate costs of outsourcing bids up front.

2.2.3 Resource sharing risks

Intellectual property and privacy concerns. One the one hand, outsourcing is a good way to share resources like talented professionals, new geographic area and limited resources. However, on the other hand, when asking for help from the outside, CIO has to pay immense attention to intellectual property and privacy concerns. This can be extra stressful consideration when CIO is employing someone from another country who does not abide by the laws of CIO's own country. Outsourcing software development inevitably make vendor access to business information and customer data, patents and proprietary technology. The moral hazard of outsourced partners will

[1] Source from: https://www.fullcontact.com/blog/hidden-costs-of-outsourcing/

be reflected in the activities of financial corruption, mismanagement and other consequences of outsourcing activities. It would be troubling, for example, if one of the programmers from vendor steals the code from the bank. How will CIO take legal action then? If vendors are dealing with patents, copyrights, or top secret information, outsourcing can be difficult. So if CIO ignore the protection of the business information, it will have disastrous influence on outsourcing activities or even the bank.

2.2.4 Management risks

1) Loss of control. If outsourced partners are working remotely, monitoring them could take time and effort. Even if there are management tools that CIO can use, there is no guarantee that an employee is doing the job right until CIO get the deliverables. Besides, CIO should also be aware of the "4x Rule" in the outsourcing activities. That is, expect work to take four times longer to complete because of possible miscommunication, inaccuracies in implementation, etc. If work is done in-house, monitoring would be consistent and corrections can be done right away. More tricky, sometimes scam artists disguise themselves as vendors. As such, it's important for CIO to verify the identity of the vendors before hiring them.

2) Communication challenges. Firstly, language barrier is always a major barricade when dealing with international organizations. Not only is a language barrier an annoyance, but it can cause fatal misunderstandings regarding key specifics of projects. Software programming may be an international language, but programmers across the world aren't necessarily multilingual. Another thing must be mentioned is that, cultural shock is significant in global or distributed software development. Software development is a human-centric and socio-technical activity. Complex interaction of attitudes, beliefs and behavioral norms by project participants with greatly different values may give rise to misunderstanding and misinterpretation of purpose that may lead to conflict, mistrust, and underutilization of professionals.

3 Describe the approach you would take as CIO to outsource part of your agile SW Development. Ensure that your approach mitigates the risks you put forward in Q2).

According to the risks put forward in Q2, we can find that outsourcing software development is a big challenge. How to meet the challenge and mitigate the risks mentioned above? Acting as a CIO of the bank, it may be a reasonable approach that completing this complex task into two steps: 1) the first step is to identify the type of the project; 2) and then based on the project type, divide the management task into three dimensions: manage the what - project scale; manage the who - project participants; manage the how - project process.

3.1 Identify the what - the type of the project

Based on the user requirements delivered by the business stakeholders or customers, firstly, we need to have an overview of the desired application including its business purpose and problems to be solved as well as a definition of success. And then we need to develop a preliminary but not overspecify project backlog - prioritize product features and group them into milestones taking into account both market and technological constraints involved. It is a foundation for the outsourcing software development.

3.2 Manage the what - project scope

1) Specify project scope step by step can mitigate the time risks and financial risks. With the agile approach, we develop the project scope step by step concurrently with development. As we progress with development, we can gather business stakeholders or customers feedback, verify our initial blueprint and modify our product backlog accordingly. If the project scope is large, we can use the "milestone budget" approach to simply breaks a larger project into smaller, more manageable projects for which it is easier to establish time and costs for the work to be delivered.

2) Scope division can help to mitigate the resourcing sharing risks. Scope division is about responsibility and ownership when some work is to be done in-house and some at the vendors. Without clear responsibilities, resource sharing risks like intellectual property and privacy concerns may be resulted. It is important that each party know what their role in the project is. To have a proper scope division and mitigate the risk of information leakage, we should clearly identify the entire scope of the project and the scope for the vendor. We can keep parts of the project independent from each other so that each team can make progress without depending on other teams to get things not only done, but done the way they should be.

3.3 Manage the who - project participants

1) Keep interaction with customers/business stakeholders. In many cases, agile teams focus too much on the safe and functional activities. The result is often a product without a market fit or a product complicated by unrelated features. Smith and Reinertsen(1998) said that, "to avoid such risks, clearly specify the role of the customer and/or user in your software product development project." If we are working on a new application with high market risk, struggle for intense collaboration with prospective customers throughout the process. Involve them in the specification process from the beginning before any decisions are made. Test our assumptions about the features and functions our customers are seeking. The customers' feedback on iterations and deliverables will be critical in prioritizing work on bug correction and the next features to be added. And if we are adapting an application just to improve its performance, and customers'/business stakeholders' availability is not an issue, we can limit their involvement in the process.

2) Keep in-house team under control can mitigate management risks. Before we outsource our software projects, we should make sure our in-house team is both capable and willing to outsource. When outsourcing part of our agile SW development, we need in-house skills to actively manage the front-end of the process focused on collaboration with our customers/business stakeholders and ensure product management.

3) Select a vendor in a suitable location can mitigate time risks and management risks. Firstly, the time zone differences have a definite impact in the consideration of the vendor's location. If the project requires much communication, it is important for us to confirm how the vendor's workday overlaps with our own. And if we need much face-to-face collaboration, the vendor in the domestic or neighbouring country may be a proper choice. Secondly, the location can also determine the vendor's ability to speak English or any other language we need to communicate in. Thirdly, culture is also an important thing to remember when choosing a location. In general, it is much more efficient to develop our product with those who have previous experience within the bank industry and the type of solution we are trying to develop. This principle can help us to mitigate the time & management risks mentioned above.

3.4 Manage the how - project process

1) A good team structure can mitigate time risks and management risks. To ensure both technical and market successes, we need two key roles in our team. One is the in-house product manager and another one is the technical leader. Firstly, the in-house product manager is responsible for day to day interaction with the development team and, therefore, making sure the outsourced partners stay on track and, ultimately, that the product is delivered. Secondly, the technical leader can be in-house or from the outsourced partners. Technical leader is responsible for technical advice and the programming. Technical leader should ensure product run smoothly at the back-end of the process. If this person is from vendor, he/she should be responsible for the delivery of the product on vendor's side. In brief, this team structure can help to mitigate the management risks and avoid CIO's personal time to be occupied.

2) Using a set of tools can mitigate the management risks, time risks and finance risks. To execute effectively and mitigate the management risks, we need a set of tools to control our project process, schedule and cost as well as ensure effective communication among the participants. There are three kinds of tools we can consider: software development tools, project management tools and collaboration tools. Firstly, software development tools include those which manage the requirements, control the source code and track bugs and improvements. For example, Mockup Builder, Github and Cloud9 IDE. Secondly, project management tools, such as Redbooth, Asana and Planbox, are used to keep tabs on the progress of the project and time and budget. Thirdly, online collaboration tools, such as Flowdock, GoToMeeting and Slack, allow the members of each team to communicate and work together.

In summary, there is no one common approach with which to handle all software development projects. To mitigate the multiple risks mentioned in Q2, we need to: 1) first, identify the project type; 2) And then manage the project scope, participants and process. The success will depend on how well we plan, organize, execute and manage the specified aspects. All in all, we need to adapt our approach to match the specific project's characteristics.

References

[1] Segue Technologies(2015). *"What is Agile Software Development?"*

https://www.seguetech.com/what-is-agile-software-development/

[2] Apiumhub(2017). *"The benefits you get by doing agile project management"*

https://apiumhub.com/tech-blog-barcelona/benefits-of-agile-project-management/

[3] Stephen Haunts(2014). *"Advantages and disadvantages of agile software development"*

https://stephenhaunts.com/2014/12/19/advantages-and-disadvantages-of-agile-software-development/

[4] Clifton Kussmaul, Roger Jack, and Barry Sponsler(2014). *"Outsourcing and Offshoring with Agility: A Case Study"*

https://link.springer.com/chapter/10.1007/978-3-540-27777-4_15

[5] Selleo(2014). *"A practical guide to outsourcing your software development"*

http://selleo.com/wp-content/uploads/2014/08/software-outsourcing-guide.pdf

[6] delaPlex Software(2016). *"Why outsourcing software development is about more than saving money."*

http://delaplex.com/dev/1109/why-outsourcing-software-development-is-about-more-than-saving-money/

[7] Joseph Chris(2015). *"9 pros and cons of outsourcing jobs"*

http://www.josephchris.com/9-pros-and-cons-of-outsourcing-jobs

[8] Travis Todd(2013). *"The hidden costs of outsourcing"*

https://www.fullcontact.com/blog/hidden-costs-of-outsourcing/

[9] Alok MishraDeepti Mishra(2014). *"Cultural issues in distributed software development: a review"*

https://link.springer.com/chapter/10.1007/978-3-662-45550-0_45

[10] Taiki Beaufils(2016). *"4 risks of outsourcing software development and how to avoid it"*

http://d8ii.com/offshore-outsourcing/risk-outsourcing-software-development/

[11] Smith, P.G. and Reinertsen, D.G. (1998). *"Developing products in half the time: new rules, new tools"*. John Wiley & Sons, Inc. New York.