Requirement Analysis and Agility

Dirk Bechtold
E-Business Beratung Bechtold
Salierstr. 5, 76137 Karlsruhe, Germany
email: bechtold@e-3b.com

Abstract

This paper is based on experiences made during many years of project management within a specific sector of the German software industry.
Typically, the software to be developed is not to replace an existing system or to reflect an existing workflow, but rather to create a totally new workflow within the organisation.
With the novelty, a number of specific problems occur. These problems have impact on the requirement analysis and the specification process, as well as on the development and implementation phase of the project.

Introduction

Customers need precise figures concerning the budget for a software project. And they need this information in advance of a project, not afterwards.
This is a basic requirement and a decisive criteria for obtaining the order confirmation in most software development projects for industrial customers.
But where to start?
The user problem, the project motivation might be well defined. But many projects start with a requirement paper of the customer and some more or less significant use cases. It is difficult to define requirements for a software solution to be designed for a completely new workflow.
It seems absolutely common that customer requirements refine, change or come up during the software development process. The outcome of a software project has minimal similarities with the system designed in advance.
Does it make sense to conduct a requirement analysis at all? Are the results of such an analysis valid?
It seems to be quite simple. All we need is a translation process, which translates the customer ideas of a software solution into software code and into a real running system.
What is so difficult?

Babel and Chinese Whisper

First of all, most developers have a lack of understanding about the business requirements of their customer. Their look at software is radically different than the customers' perspective. It is inside-out, whereas a customer looks at software as a business tool.
Customers and software engineers don't speak a common language, they don't understand each other.
So we need translaters:

1. The Business Analyst, he will understand the business case of the customer and supports him writing the requirement specification paper
2. The Software Designer. He tries to translate the results of the business analysis into an architecture and writes the functional specification
3. The Software Engineer. He will write code, in order to realize components or modules, specified through the functional specification paper
4. The System Integrator. He will integrate the developed modules into the overall software solution.

It is quite obvious, that all those roles need to have the same understanding of things, speak the same language together.
The more interfaces exist, the more likely the result departs from, what the customer originally expected.
We all know the Chinese whisper phenomenon.

Can we finally start?

Second, the project stakeholders on the customer's side tend to be rather impatient during the analysis phase of a software project. The reason is: the preliminary time of a project can last from several months to a year easily. From their perspective, they already spent too much time during the bid phase, comparing offers and deciding which company is winning the bid to develop the new system.
From their point of view, they wasted hours and hours with internal planning meetings, convincing internal opponents with project presentations. Finally calculating the "ROI" to convince the CEO. Now, the project is approved and appointed, contractual issues are settled, so lets start!
What about functional requirements, systems architectural restrictions, data modelling, rights and roles, input/output dialog fields?
In fact there are still many open questions when a project is about to take off. Everybody knows, moving targets are difficult to aim at and almost impossible to hit!

Moving targets and the agile family

Here comes XP, Scrum and all brothers and sisters from the agile family. They are pointing out:
No problem with the moving targets, we still have time within the development process to find out, where the journey goes.
In fact it is true what the agile family members tell us: There are plenty of requirement analysis papers, specification sheets, design documents, which didn't even come close, to what will be implemented at last. Requirements always change, because customers discover new features they urgently need, or modify features, which were thoroughly defined within the requirement phase.
So from the development point of view, it seems to make sense to execute the requirement process on the fly, rather than before.
From the customers point of view, this proceeding is rather dubious. Consequently, it is not possible to generate a reliable fixed price offer during the bid phase. And honestly, an expense related deduction of software development projects is rather exotic.
Advocates of agile software development neglect the fact, that at least the master of budget has absolutely no interest to acquire a project which is a financial black box. Although the flexible iteration process allows to steer the project output close to the actual requirements of the customer.

Here comes the Waterfall Model

The supporters of traditional project management methodology will argue: This degree of uncertainty is absolutely not necessary and can be easily avoided. Project sequences are the solution. Requirement analysis first. Then design, and then, after extensive specifications it will be possible to obtain certainty on how much effort is necessary to build the system.
It should be rather plausible towards the customer, first to offer the analysis phase based on a fixed price. The implementation phase and the expenditures depend on the results of the analysis phase. Therefore, this professional procedure to give estimates for the implementation phase after conducting the analysis phase, is absolutely convincing.
And for the case, although everything is completely specified, the customer suddenly brings up new ideas and requirements, a formal change request process will be set up in order to solve these problems.
Unfortunately, competitors seem to know a way to give fixed price offers for the whole project, without conducting explicit analysis phases.
In many cases, this is the knock out criteria for "analysis-phase-first" offers.
But how does a fixed price offer for the whole project arise?
Experiences from recent projects provide a base for realistic estimation of expenses. But per definition, each and every project is different and therefore not exactly to calculate. An extra charge for uncertainty can reduce the impact of wrong calculation of the offer. But no more.
So this can not be ideal procedure neither.

Another side blow from the agile front

The more you analyse and specify up front, the more change requests you will gain during the project, that is the provoking correlation thesis from the agile fraction. And there is some truth in it!

Where is the general approach?

How to execute the project successfully, in terms of customer satisfaction (i.e. time, budget, quality)?
The answer is as simple as unsatisfying: There is no general approach!

Usually, the satisfaction of the customer is related to the level the objectives of the project that have been achieved. These objectives differ from customer to customer.
From customers' perspective, the agile approach seems to burden most work on their shoulders, and it appears, that the contractor does not like to commit himself, neither to delivery dates, nor to the scope of delivery.
Although, there seems to be a correlation between the degree of novelty of the software system or the supported workflow and the acceptance of flexible integration of requirements throughout the project.
The closer the customer is linked to software and software products, the higher is the acceptance to allow an iterative approach within the project.
It is important that even a highly agile iteration process should include at least one, in fact, highly un-agile element, that is documentation!
The lack of imagination on the customer side, compared with forgetfulness on both sides are natural enemies of agile adjustment processes.
"But that's exactly how we discussed the feature at that time. Now the customer wants it vice versa!" This sentence, furiously spoken, can be heard in many agile projects.
In deed, the lack of imagination, together with missing documentation, lead me to the recommendation: "Document the iteration plans and specify the features in writing and adjust them with your customer."
We still face the problem of different languages, but at least we eliminate the annoying and unnecessary factor forgetfulness.

Conclusions

- Get as much specification up-front, as you need in order to make a reliable estimate for the expenditures of your project.
  This avoids a miscalculation and so an unrealistic offer, which might either ruin your company or your reliability towards the customer
- Never be too sure that you really understood what he wants, unless you finally receive the acceptance test signature from your customer.
- Take the flexible approach to design the solution. During the project, include validation sessions into your iterative process, in order to refine the requirements of your customer.
- The degree of agility will see its natural limit by the customers willingness and ability to be involved into the process.
- No matter how agile, make sure, that vital results are documented in writing
- Don't limit yourself on customizing your products. Customize your process to your customers' needs as well.