REQUIREMENTS ENGINEERING ISSUES IN AGILE DISTRIBUTED SOFTWARE DEVELOPMENT

Isu Kebutuhan Teknis dalam Pengembangan Perangkat Lunak Terdistribusi Agile

Mohammad Anggasta Paramartha
Bank Indonesia (Kantor Pusat)
Jl. MH. Thamrin 2 Jakarta 10350, Telp : 131 (local fare), 1500131 (outside Indonesia)
Fax (+62)21 386 - 4884
E-mail: anggasta@gmail.com , anggasta.i@bi.go.id

Naskah diterima tanggal 3 April 2017, direvisi tanggal 31 Agustus 2017, disetujui tanggal 15 September 2017

Abstract

There are two approaches of software development and their combination that have attracted a lot of interest from the research community lately. The first is the Distributed/Global Software Development, which entails development in multiple geographically dispersed sites. The second approach is the Agile Software Development, which incorporates an evolving development process for better adaptation to changing environments and requirements. Their combination is a challenging topic due to a lot of contradicting characteristics. In this paper, the successful communication between remote sites and especially the communication of requirements in an Agile Distributed Software Development process are investigated. The research is based on a case study at Cegeka, a Belgium ICT company with branches in the Netherlands and Romania. The Dutch and Romanian sites are engaged in agile global software development practice, facing a situation with requirements understanding. Enhancement of the awareness of the vision of the product and the vision of the company through the communication of requirements between the Business Analyst and the Scrum Master is the main challenge that this paper aims to address. However, due to the limited information we have on how the Scrum Master and the Business Analyst from Cegeka communicate, we cannot give very specific answer but just a general solution and best practices. Nonetheless it should be a good starting point for improving requirements communication within the distributed software development process between the Romanian and the Dutch sites of Cegeka.

Keywords : Agile, Scrum Master, Requirements Communication, Distributed, Software development

Abstrak

Ada dua pendekatan pengembangan perangkat lunak dan kombinasi dari kedua pendekatan tersebut menjadi bahan yang menarik banyak minat komunitas riset akhir-akhir ini. Kedua pendekatan dimaksud yaitu pengembangan perangkat lunak metode Distributed/Global, yang digunakan untuk lokasi yang tersebar secara geografis, serta pengembangan perangkat lunak metode Agile, yang dapat dengan mudah beradaptasi terhadap perubahan lingkungan dan kebutuhan. Kombinasi kedua pendekatan tersebut menjadi tantangan karena banyak karakteristik yang saling bertentangan. Dalam tulisan ini, diamati proses komunikasi yang berhasil antara dua daerah yang berjauhan, utamanya terkait kebutuhan dalam pengembangan perangkat lunak agile distributed. Penelitian ini merupakan studi kasus di Cegeka, sebuah perusahaan TIK milik Belgia yang memiliki cabang di Belanda dan Rumania. Daerah Belanda dan Rumania telah mencoba praktik agile global software development, namun pemahaman akan kebutuhannya belum sama. Peningkatan kesadaran akan visi produk dan visi perusahaan melalui komunikasi terkait kebutuhan antara Analis Bisnis dan Scrum Master adalah tantangan utama yang ingin disampaikan makalah ini. Namun, karena terbatasnya informasi terkait bagaimana Scrum Master dan Analis Bisnis dari Cegeka berkomunikasi, penelitian ini tidak dapat memberikan jawaban yang sangat spesifik namun solusi umum dan praktik terbaik. Meskipun demikian, hal ini menjadi titik awal yang baik untuk meningkatkan komunikasi terkait kebutuhan dalam proses pengembangan perangkat lunak agile distributed antara Rumania dan Belanda di Cegeka.

***Kata kunci**:* Agile, Scrum Master*, Kebutuhan Komunikasi, Terdistribusi, Pengembangan Perangkat Lunak*

INTRODUCTION

Over the last few years, there has been an increasing interest from software enterprises towards engaging in Global Software Development (GSD). According to Hashmi et al. (2013), GSD involves development of software in a distributed environment which crosses multiple geographical borders. GSD can take two forms, outsourcing and distributed teams within the same organization that are scattered in different countries (Layman et al., 2006). Some countries even promote themselves as main software outsourcing destinations, for instance India, China, or Eastern European countries.

On the one hand, developing a software product globally brings many advantages such as reduction of development cost and less overall project costs, access to a large pool of knowledge, skills and labor (Carmel & Agarwal, 2001; Layman et al., 2006; Hashmi et al., 2013). On the other hand, the distance between teams brings communication, coordination and control problems (Carmel & Agarwal, 2001). Lack of trust is also an issue (Ramesh et al., 2006) and it could be considered as an influential factor for the low level of understanding "of the overall context or background information at distant sites" (Herbsleb & Mockus, 2003). Additionally, as Holmström et al. (2006) point out, geographical distance can hinder the communication of vision and strategy among distributed teams.

Apart from the distributed software development hype, agile software development has been accepted widely as the new paradigm of software development. In contrast to the traditional waterfall model, where integration changes, e.g. design issues, interface errors or performance issues, are considered complex and a driver for higher costs of change, Agile promises easier adaptability in changes which contributes to increasing quality of software products (Royce, 2009). Among the principles that govern the Agile software development the collaboration aspect is prevalent both in terms of regular communication with the customer for adjusting the priorities, scope and plan of the project as well as in terms of teamwork among distributed development teams (Royce, 2009). Agile method can support multi region and geographic location with effective communication between team members (Dorairaj et al., 2011).

As a result of its adaptability to change and its evolving scope, agile development offers a solution closer to customer's needs. One of the agile software development methodologies widely adopted nowadays is Scrum, which comprises a project planning methodology for managing and tracking software development and offers a shared vision and awareness of project activities (Holmström et al., 2006; Hossain, 2008).

Taking into consideration the benefits offered by both agile and distributed or global software development, it can be beneficial to combine them into practice with the expectation that greater advantages will be obtained. Successful integration of the characteristics of GSD and agile development is essential for reaping the expected benefits (Hossain, 2008). Despite the benefits, this combination might also bring more complexity and challenges to tackle. An area where specific focus should be addressed is the communication of requirements in such a distributed and agile context. The evolving quality requirements (Ramesh et al., 2006) and the effectiveness of the requirements' handing-over (Hashmi et al., 2013) are challenges that need to be confronted.

To date there has been little best practice and academic literature covering the topic of development process issues in agile GSD (Hossain, 2008). In this paper, an analysis is made on the issues related to the requirements engineering process in an agile distributed context. The analysis in this report is based on the practices followed and performed in the Dutch branch of a Belgium software company namely Cegeka. The incorporation of the vision in, and the communication and understanding of requirements in dispersed teams are challenges investigated in this company setting.

The analysis is made based on literature research.

The next section presents the current situation and problem confronted by the company. In the third and fourth sections, the research questions are defined and the research methodology that was followed is described respectively. Later on, a literature review is presented as well as the findings concerning possible solutions for addressing the requirements issues of vision, understanding and communication. In the sixth section, based on this literature review, we give our own recommendations to the company. Then, in the last two sections the limitations of this report and the conclusion which summarizes the findings and contribution of this research are provided.

Case Study

In this section the company is presented as well as its current way of working. The information has been acquired from the website of the company and from discussions with Mr. Gerard Murre (Director of the Shared Software Factory - Netherlands) and Mr. Laurentiu Oprea (Business Unit Manager - Romania) who are involved in the process of the software development.

The Company

Cegeka is an ICT (Information and Communication Technology) company founded in Belgium in 1992. They provide full range of ICT services such as application development and integration, outsourcing, consulting, Infrastructure-as-a-Service and Platform-as-a- Service. Their branch in the Netherlands focuses on the health care sector and the social living sector of the Dutch market. They offer standardized software solutions which can still be tailored to a certain extent to match customer needs and specifications as well as "availability, capacity and flexibility" (ICT Outsourcing Services). In general, though, they aim at addressing the needs of a mass market rather than a specific customer.

Through some acquisitions in Romania, they developed their subsidiary which is responsible for the software development. 95% of software development is done in Romania. The selection of Romania as an outsourcing development site was based on strategic decisions concerning its cultural and geographical proximity, e.g. approximately same time zone, near-shore location which is quite easily accessed, and the availability of skilled developers with low cost.

The vision and the mission of the company are stated in the Table 1.

Table 1. Vision and Mission of Cegeka

Vision	"ICT can give you a strategic advantage. This is only possible if your ICT fits in seamlessly with your business. Cegeka wants to work together with you to ensure that your business and ICT remain permanently harmonised.
Mission	"We want to help you realise your ambitions, interpret your needs and solve your problems by providing you with high-quality ICT solutions that make the difference."

Agile Global Software Development at Cegeka

Cegeka follows agile software development process in distributed locations. Agile global software development was promoted by Cegeka Belgium 5-6 years ago, whereas in the Netherlands it has been applied only for the last year. Because of the longer experience in agile software development, the Belgian head office can be considered, which was also admitted by the interviewees, more mature in deploying this methodology than the Dutch branch. Scrum is used as the method for managing the agile development process. Effectiveness of the overall process is a key prerequisite which is currently lacking in the Dutch-Romania joint way of working.

According to the information provided by Mr. Gerard Murre and Mr. Laurentiu Oprea during the interview, there is a certain composition and distribution of roles among the team members participating in a project. Cegeka project teams are composed by a Business Analyst, who has also the role of the product owner and the customer proxy, a few developers (the number of developers depends on the project), testers and a Scrum Master who is the Agile project manager. Only the Business Analyst is located in the Netherlands, while the rest are in Romania. The reason behind the decision to locate the Business Analyst in the Netherlands is to have better understanding of the Dutch market and regulations related to the target sectors and to which compliance should be taken care of.

Following the (Cegeka's Agile Software Factory) documentation on its way of working, the responsibilities of the different roles in the project team are explicitly defined. So, the responsibility of the Business Analyst is to have a close communication with the customer, define, clarify and prioritize its needs and requirements, eliminate possible assumptions and fill in the requirements backlog. For defining the requirements, user stories and acceptance criteria are used, which are worked out in collaboration with the customer. The requirements are then communicated by the Business Analyst to the Scrum Master and consecutively to the development team, which is responsible for the development, the technical design and the architecture of the solution. Finally, the Scrum Master assists the team in working in an agile manner, monitors the project risks and eliminates or mitigates possible impediments. The project teams work in two- week sprints using sprint backlogs in which the user stories are divided in tasks, but they present their progress in daily scrums through videoconferencing meetings.

An essential component of this process is the Requirements Management. According to (Cegeka's Agile Software Factory), the requirements, before being stated in the form of user stories, are analyzed by the customer with the assistance and supervision of the Business Analyst. This procedure results in a High-Level Analysis document which includes a description of the business processes, a functional description of the application and an initial product backlog. The High-Level Analysis document is updated, refined and modified during the development process, but its purpose is to provide an overview of the business processes to which the solution will contribute. In every iteration that follows the initial project and requirements setup, a more detailed analysis and representation of the requirements is applied, leading to user stories. Both processes are under the responsibility of the customer proxy, who in the case of the Dutch-Romanian collaboration is the Business Analyst.

Problem Statement

During the discussion with the Director of the Shared Software Factory and the Business Unit Manager, some issues emerged showing that the distributed collaboration and development between the Netherlands and Romania face challenges. It was valued as highly important by both persons that the developers in Romania feel attached to the company and understand the company's vision and strategy.

Having team members in two different countries requires efforts for establishing and maintaining good communication among the distributed team members and understanding of the business processes and reasons by the whole team. The main problem identified through the interview is that the developers have a lack of understanding of the company strategy and vision. This is mostly imposed by miscommunication between the Business Analyst and the Scrum Master. The Scrum Master functions as the intermediary in the information flow, the gatekeeper. The information (requirements) is generated from the face-to-face collaboration between the customer and the Business Analyst. Afterwards, it is transferred to the Scrum Master, who in the end explains it to the

developers. The whole process is depicted by Figure 1.

The only link between the developers and the Dutch branch is the customer's requirements; this is the piece of information that should be handled properly. Successful and effective communication of requirements can be a way of accomplishing the company's wish to increase the level of awareness among developers from a business perspective. Thus, the purpose of this paper is to find a solution to fill in the gap between the two parts of the team by focusing on the requirements and communication aspects.

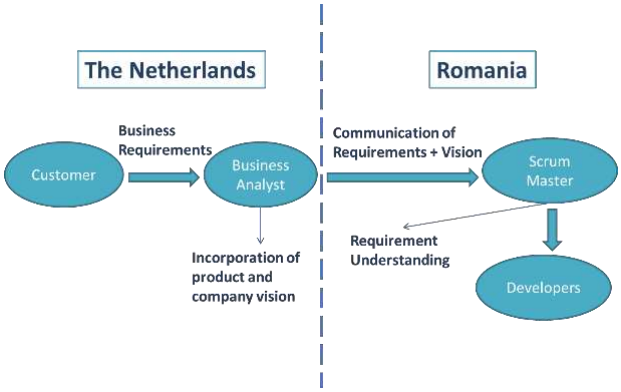

Figure 1. Requirements transfer process from Cegeka Netherlands to Cegeka Romania

Research Questions

Based on the results from the interview, we formulated the main research question as:

> *How to improve the requirements communication between the Business Analyst and the Scrum Master so that the developers have a better understanding of the company and product vision?*

The following sub-questions are then derived in order to provide guidance for answering the main question:

1. How to incorporate company and product vision into the software requirements or through the whole agile process?
2. How to make communication between Business Analyst and Scrum Master better?
3. How to make developers more aware of the company and product vision?

Research Methodology

Two ways of approaching the topic have been conducted. Firstly, to obtain a better understanding of the current situation of Cegeka, we conducted an interview with Mr. Gerard Murre (Director of the Shared Software Factory- Netherlands) and Mr. Laurentiu Oprea (Business Unit Manager - Romania) at the Cegeka office in Veenendaal. From these discussions, information for the company, the current way of working and the present issues were gathered. For further clarification on the topic, e-mail contact was used.

Secondly, a literature review is chosen to gain knowledge and a better overview of the areas of concern in order to cover the three sub-questions. In literature review, "the researcher is concerned with charting the development of a set of ideas, and with placing them within a descriptive framework" (Cornford & Smithson,2006). The scope of the research is the investigation of available processes for appropriate handling of requirements in agile distributed environments in the software industry. Therefore the following steps are taken:

1. Searching and choosing related literature
2. Understanding Requirements Engineering (RE) and scoping it down to agile RE and Goal-Oriented RE
3. Identifying ways for incorporating the company vision in the requirements
4. Investigating the requirements understanding and its importance
5. Pointing out applicable and suitable techniques for the translation of requirements from a business level to a technical level
6. Identifying possible solutions for the issues addressed by the sub-questions

In the last section of this paper, the results found in the literature study will be combined with the information from the interviews to come up with suggestions for the company that might be useful for tackling the identified issues.

Literature Review

To support the source of the misalignment and misunderstanding, firstly, a justification of the current situation in Cegeka based on research papers is provided.

The decision of the company to divide the team members' responsibilities based on their location is supported by the literature, since as Hashmi et al. (2013) mention the "onsite team works closely with the client in order to elicit their business requirements" and "the requirements gathered and managed by the onsite team are handed over to the offsite team so that software development can be carried out." This requirements handover, though, entails risks because of the geographical distance and the communication barriers (Hashmi et al.,2013).

Additionally, the fact that the company relies its services on standard solutions and its target is not to develop fully customizable products that directly suit the wishes and needs of a specific customer but of a market segment, implies that the market-driven software product development (MDPD) approach is followed. According to Fogelström et al. (2010), in such an approach of software development practice, the development organization decides what functionality should be delivered to a market segment. Partially customer-specific solutions cause a misalignment with the key principle of agile development, which emerged in order to achieve higher customer satisfaction by fulfilling its needs. So, "application of agile properties in an organization operating in market-driven context places limitations on product management activities, and may have a detrimental effect on long-term product development" (Fogelström et al., 2010).

Another factor of the agile practices that is considered necessary in Cegeka is the personal attributes of the members comprising the development team as well as the supporting management. In agile software development, self-managing teams and a leadership-and-collaboration style of management, where the project manager is a facilitator (Hossain, 2008), are main factors that influence the success of the project.

In particular, soft skills as communication capability and collaborative spirit, and self- discipline are traits mentioned by the Business Unit Manager during the interview. Thus, the dependency on personal traits and the necessity for continuous collaboration may also influence the communication and understanding of the requirements, which is the central communication medium between the developers and the Business Analyst.

So, there are three areas in the way of working that is applied by Cegeka which are vulnerable to possible obstacles, the requirements understanding and handover, management activities during the software development process and communication as a top-layer covering the whole development process.

Requirements Engineering

Requirement can be defined as a property that a product must have in order to provide value to the stakeholders (Wiegers, 2009). Thus, in software engineering context, the software requirements can be considered as the foundation for software quality. Requirement Engineering is a subfield of software engineering dealing with identifying, modelling, communicating and documenting the requirements for a system (Paetsch, Eberlein, & Maurer, 2003).

Within the process of requirement engineering, there are several key activities involved: Elicitation, Analysis and Negotiation, Documentation, Validation, and Management (Kotonya & Sommerville, 1998). The end goal of requirement engineering is to make complete, consistent and relevant requirements. By implementing a high-quality requirement engineering process, some benefits can be achieved such as faster development time, reduced development rework, lower costs, fewer miscommunication and higher customer satisfaction (Wiegers, 2009).

Software system requirements are generally classified into functional requirements and non- functionalities that the system should and should not provide, the response of system for specific inputs and the behaviour of the system in a specific situation. By understanding the functional requirements, developers will understand what they need to build into the product to enable users to achieve their goals. (Sommerville, 2007).

On the other hand, non-functional requirements define constraints on the services or functions provided by the system (Sommerville, 2007). Non-functional requirements mostly apply to the whole software system as global qualities such as flexibility, maintainability, or usability (Mylopoulos, Chung, & Yu, 1999). However, non- functional requirements are often quite hard to be implemented and validated. Failure in meeting this type of requirement can lead into unusable software system.

According to (Sommerville, 2007), non-functional requirements can be further divided into three types: product requirements, organizational requirements and external requirements. Accordingly, incorporating the company goals along with the product goals into the software being made falls into non-functional requirements. This goal-oriented requirement analysis (requirement engineering and requirement analysis are used interchangeably in RE literatures) puts emphasis on the description and evaluation of system design alternatives to capture their relationship with the goals of organization in a software development project. Using this approach, it is expected that the software requirements process will be more thorough, complete, and consistent. (Mylopoulos, Chung, & Yu, 1999)

The next part will elaborate more on the relation between company and product vision with software requirements in agile software development context.

Visions and Requirements

According to (Qumer & Henderson-Sellers, 2008), Agile methods are welcomed by both managers and programmers as providing a more needed release compared to traditional software development approaches. Nonetheless it could be inappropriate for companies to be fully agile in all aspects of developments; they should retain well-known and trusted elements of a more traditional approach within an overall agile project. Indeed the absence of a shared vision between the business and the development parts is one of the main factors of software project failures (Qumer & Henderson-Sellers, 2008) and the business-agile alignment bridge is an issue that has not been investigated in detail by the agile community.

For (Vähäniitty & Rautiainen, 2008) three key words are linked together: Vision, Product and Business Goal. For example products are software that the company is developing. They should contribute to a vision. A vision describes the “grand plan” for one or more Products, and is concretized as one or more Business goals. The framework proposed by (Vähäniitty & Rautiainen, 2008) is illustrated in Figure 2

According to (Pichler, 2013) as shown by figur 3, agile product planning is composed of three levels: vision, product strategy and tactics. The vision is the overall goal, the product strategy the path for reaching the vision, and the tactics the steps for achieving this goal. Whereas the vision is caught by a short statement, the strategy communicates different aspects including the markets or market segment targeted. The tactics go deeper by describing the product details using user stories, design sketches, scenarios and storyboards.

.

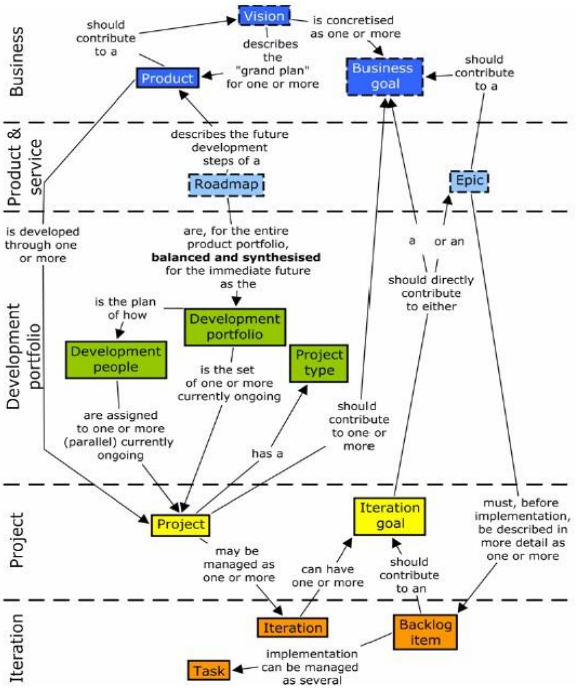

Figure 2. Linking product and business planning with agile development

Figure 3. Three levels of product planning

The Product Strategy

The product strategy is the bridge between the business strategy and the product development (Rautiainen, Lassenius, & Sulonen, 2002). It incorporates a long-term view to product and technology planning. The overall strategic ambitions and goals of the company should be taken into consideration. The practical product manager serves as a hub of market and product information, he or she works closely with Development, Marketing, Sales, and other departments (Pragmatic Marketing & Enthyosis, 2012). The product management team is a key executor of the strategy. The team will translate corporate strategy into product strategy and will create roadmaps that drive the work of The company's employees (Thomson, 2012).

Figure 4. Product Strategy Diagram

As illustrated in figure 4, a product strategy contains:

- Business objectives
- Descriptions of target markets, based on results of market Research
- Results of research about potential clients and their needs
- The way the product should be viewed by clients
- Product features and benefits
- Selling strategis
- Comparison of the product features and pricing with competitors 'ones
- Product changes that might enable better market positioning of the product

Product Management: Product Owner and Product Manager

According to (Pragmatic Marketing & Enthyosis, 2012) there is a common problem in companies that could explain the lack of vision of developers. Indeed by adopting agile development methods, new planning methods and new roles are created. This is the case for the Product owner who is responsible for providing customer and market information to the team. Nonetheless product owner and

product manager are not the same. In fact a product owner's responsibilities are just a small part of product management. The role of the product owner includes retaining and prioritizing the product backlog including specifying and collecting individual user stories (Singh, 2008).

Product owners can fill in the gap between a product manager's role which is to understand the needs of the marketplace, and the development team's need for product direction (Pragmatic Marketing & Enthyosis, 2012). Nonetheless there are a few characteristics that will differentiate a product owner from an experienced product manager. Good product managers look across the product line for ways to make the overall collection more valuable. But the product team focuses on local optimization: what is the best for the release plan with little consideration for portfolio-level need. Without some strategies present in every sprint-level prioritization, the company loses many opportunities to profit through product bundling.

Because the closest equivalent to product owner in most companies is the product manager, it seems natural to equate the two (Pragmatic Marketing & Enthyosis, 2012) which is a mistake because the product manager has more knowledge about the strategy of the company compared to the product owner.

According to Gottesdiener (2009) three levels of requirements can be identified:

- During product road mapping workshops, The main goal is to probe the "big view" of requirements to build a strategy for the entire product.
- In release planning workshops, the time horizon is smaller but is used to get a "pre-view" of requirements for the next release.
- In iteration planning workshops, the "now- view" is explored. It is composed of plans of small and concise set of requirements for the instant sprint

A roadmap outlines what the team plans to do. It is the vision of the project, but the team can still make corrections to the plan. The product roadmap is vital especially in large and complex product (Gottesdiener, 2011). It is not necessary to know each specific route, but the overall way must be clear.

The key deliverables for the product roadmap workshop, in term of requirements, are the vision statement and the product roadmap. The product vision statement is a short summary for communicating in what way the product is linked to the company's strategies (Layton, 2012). The vision statement must articulate the goals for the product.

According to (Turk, France, & Rumpe, 2002) ensuring that the distributed team members all preserve the same vision is possible with a good documentation of requirements and designs. Products management deliverables such as market segments and competitive positioning can also be integrated to harden the product strategy (Gottesdiener, 2011). According to (Morrison, 2009), the roadmap should be used as a communication tool. It is absolutely necessary that product managers constantly communicate. The roadmap can be used as a good communication tool to communicate to:

- Developers, Test Analyst and the wider technical team.
- The line manager and heads of departments
- Managing Directors and Chief Executives

Requirements Communication Between Business Analyst and Scrum Master

Requirement engineering (RE) approach in agile software development environment is different with the traditional one. Agile RE aims to convey the customer requirements to the developers without making extensive requirements documentation through formal requirements analysis and design phases. Instead, the requirements arise throughout the development process based on feedback from stakeholders. On the contrary with the

traditional RE approach in which the customer is only involved at the beginning of the project, in agile RE process the customer is involved throughout the whole agile software development project. (Cao & Ramesh, 2008)

In practice, it might not be possible for customer to interact directly to the developers, especially if they are geographically separated as in distributed software development project. In that case, the customer will discuss the business requirements with a Product Owner (sometimes called Business Analyst or Customer Proxy) who is located close to the customer location. These business requirements will be then translated into user stories.

User Stories are short statements (one or more sentences) which describe product functionalities desired by the customer/user, which also connect acceptance tests, help planning and prioritizing, and enable monitoring project health (Liskin & Schneider, 2012). Good User Stories should comply with six criteria compiled into INVEST acronym (Independent, Negotiable, Valuable, Estimable, Small, Testable) as suggested by Bill Wake, the author of Extreme Programming Explored and Refactoring Workbook (Cohn, 2004).

Communication about user stories is highly important to make sure the team understands the direction of the project in order to ensure project success. To facilitate collaboration and communication in a collocated agile development team, the use of physical artefact is encouraged. Generally, two kinds of artefacts are used: the story card and the Wall. The story card is a relatively small index card in which the user stories will be written, while the Wall is an area of vertical space such as filing cabinets, flip chart, or a wall, where active story cards (which will be tackled in an iteration) are displayed according to a certain layout convention. (Sharp, Robinson, & Petre, 2009)

In the case that the agile team is distributed in remote locations, the geographical distance makes it harder for the team to collaborate and communicate the user stories. The physical artefacts used in collocated environment mentioned previously might not be useful anymore. To cope with this situation, some web- based tools and software have been developed such as Sourceforge issue tracker, Whiteboard Photo, DotStories or MasePlanner to name a few (Rees, 2002; Morgan & Maurer, 2006). By using these tools, the creation and organization of the story card can be facilitated in similar way with collocated team.

After finishing the user stories, the Product Owner will then transfer the user stories to the Scrum Master who is responsible for managing the software developers. However, even though user stories are suitable in defining the needs of the user, they do not specify how the system should response to specific inputs from the user within different contexts. This leaves room for different interpretations from the development team which might also lead to misinterpretation of the requirements. In addition, because the user stories are written in business/natural language while the scrum master and the development team are basically technical people, misunderstanding might occur because business people and the technical people do not generally talk with the same "language". This issue about understanding of the requirements will be elaborated in the next section.

Understanding of Requirements

According to (Christel & Kang, 1992) the problems of requirements understanding can be separated into three issues:

- The communities involved in elicitation possess a variety of backgrounds and experience levels, so that which is common knowledge to one group may be completely foreign to another. This makes it difficult for a requirements analyst to interpret and Integrate information gathered from these diverse communities.
- The language used to express the requirements back to these stakeholder communities may be too

formal or too informal to meet the needs of each of the groups, again because of the diversity of the communities.

- The large amount of information gathered during elicitation necessitates that it be structured in some way. The understanding of this structure is dependent on the characteristics of the stakeholder communities.

Considering the proximity of this research and constraint with respect to the information provided by the organization, this section of the research would be focused on the first and second issues. In order to comprehend requirements, according to (Paetsch, Eberlein, & Maurer, 2003) the documentation, validation and management of these requirements should be done appropriately such that the purpose of the documentation is to communicate the requirements between stakeholders and the developer.

Meanwhile, the management of the requirements is to capture, store, disseminate, and manage information. In the context of this research the tier on requirements validation (fulfilled at developers level) is not analyzed as this inquisition is focused on the communication of requirements from the Business Analyst to the developers via the Scrum Master.

The apprehension of total excellence and understanding in requirements specifications is so far understood poorly. Software metrics according to (Fenton, 1991) have mostly focused on the output of the final design or production phases, or on detailed process versification. Whereas, these accomplishments have focused more on the issue of 'building the product right' than 'building the right product', whereas both focused should be covered extensively to ensure quality from the user's point of view (Boehm, 1984).

In addition to the ensuring quality from the user's point of view, the context in which requirements understanding takes places is usually a human activity such as the programmer or developers. Therefore, requirements organization and apprehension needs to be sensitive to how people recognize and understand the setting around them, how they collaborate and how the sociology at the place of work affects their behavior.

Moreover, according to (Nuseibeh & Easterbrook, 2000) there is an important philosophical element in understanding requirements. Requirement is concerned with the interpretation and understanding of stakeholder's terms, definitions, concepts, goals and viewpoints. Hence, understanding requirements must therefore regard itself with an understanding of judgments of stakeholders, the question of what is apparent in the world, and the question of what can be acknowledged on as equitably right.

Issues as elaborated above become important whenever one wishes to discuss about certifying requirements, especially where stakeholders or leaders may have unequal missions and incompatible belief systems. The same issues being discussed also become important when selecting a preferred modeling approach, because the choice of the selected approach affects the set of phantasm that can be modeled, and may even hamper what the developer is capable of observing.

RESULTS AND DISCUSSIONS

After investigating a wide range of relevant areas concerning the requirements in software development and their understanding, refinement and handing over, a variety of propositions for improving the requirements communication has been identified. Based on them, suggestions tailored to the situation faced by Cegeka are presented:

R1. Parallel SPM and Development sprints

An agile Software Product Management (SPM) method which follows the Scrum methodology is proposed by Vlaanderen et al. (2011) as a way of improving and aligning requirements with the product vision. The main

idea of the method is to refine the requirements of the software product through a Product Management Sprint Backlog, for which both the Business Analyst and the Scrum Master of Cegeka should be responsible. The SPM sprint and the Development sprint are conducted simultaneously with a small phase difference (Figure 5).

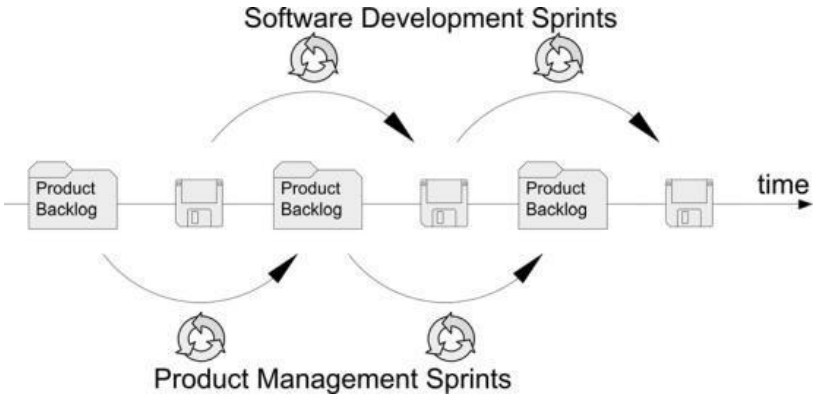

Figure 5. Alternating SPM and Development sprints

A more detailed description and explanation of the method is provided in Appendix A. So, a close cooperation of the two key stakeholders engaged in the requirements communication in Cegeka is mandatory for producing well-specified requirements in the end of the SPM sprint. Following this methodology, which combines the collaboration of the Business Analyst and the Scrum Master with the developers and the incorporation of the product vision in the requirements by enhancing their quality, can achieve a higher understanding by the developers and substantial improvement in the requirements handover process. In addition, from customer perspective, it can bring advantages to be more adaptable and responsive on business changes.

R2. Informal communication via formal channels

In order to improve the communication between the business analyst in The Netherlands and the scrum master situated in Romania, a project lead should be designated as the primary point of contact for each location and these leads should be responsible for facilitating communication across the teams. By improving communication can be also reduce and avoid gaps between requirement and development/implementation which may caused high effort in next phases.

R3. Balanced Coordination

In a typical agile development arena teams usually rely on minimal coordination of the team's activities by project managers. Consequently these project managers' coordination roles should become highly significant and important. In addition to the preceding recommendation on communication, in balanced coordination the project leads should coordinate the teams' activities going on in Romania and The Netherlands to help achieve project goals and organization visions.

R4. Constant Communications

Cegeka can implement a variety of mechanisms to maintain constant communication between the scrum master and business analyst. Short meetings can be scheduled each workday to identify issues, track project status and invite ideas and critiques. Also teams situated in Netherlands and Romania can engage in online chat extensively and the project leads (Business Analyst and Scrum Master) can be on call almost round-the-clock via any method. While this instant availability has proved to be beneficial there can at least be certain burdens.

Senior managers can use conferencing to initiate new development cycles and assess progress at the end of each cycle and discuss critical issues.

R5. Roadmap sharing

The roadmap should be used as a communication tool to communicate with directors, managers but also with developers, testers and the rest of the technical team. It is absolutely necessary that the Business Analyst and the Scrum Master constantly communicate. The product strategy should be also transferred to the whole team by the Scrum Master because there is a direct link with the business objectives and the target markets. Thus developers will have a good understanding of the product strategy and the goals of the company.

R6. Web-based User stories tools

Cegeka could make use of web-based tools to facilitate collaboration and communication of the geographically separated development team in creating and organizing the user stories. As a result, understanding about user stories can be enhanced, especially about the vision of the company and the product incorporated in the user stories.

A mapping of the recommendations (R) with the research questions (RQ) of this study is provided in Table 2 for assessing the value of the recommendations with regard to the issue faced by Cegeka as it is decomposed in the three research questions.

Table 2. Mapping Recommendations to Research Questions

Recommendations	Research Questions		
	RQ1	RQ2	RQ3
R1. Parallel SPM and Development sprints	√	√	
R2. Informal communication via formal channels		√	
R3. Balanced Coordination		√	√
R4. Constant Communications		√	
R5. Roadmap sharing	√		√
R6. Web-based user stories tools		√	√

Limitations

This paper consists of an analysis on a case study in the Dutch branch of a software company that works in an agile distributed manner for the development of the software products with its subsidiary in Romania. The analysis of its collaboration and communications issues was conducted in the context of the course "Global Software Management" in the University of Twente for the fourth quartile (April-June 2013). Thus, there was a limited time of eight weeks for the research and analysis on the topic.

Additionally, a milestone for the research was the information provided by the Director of Shared Software Factory and the Business Unit Manager during our meeting in the offices of Cegeka. The discussion was very informative and eye-opening for clarifying issues concerning the topic, but a more enhanced, complete and in depth understanding of the situation as well as a different perspective could have been gained by contacting directly the Business Analyst in the Netherlands and the Scrum Master in Romania. After considering and reviewing the initial data from the interviews, we tried to have further discussions with the Business Analyst and the Scrum Master to verify our assumptions and to clarify the actual way of collaborating and the characteristics of their communication, but we did not have the opportunity.

Whereas the information from the interviews were very useful, for understanding and gaining a more complete idea on the part of the communication between the Business Analyst and the Scrum Master concerning the

requirements, the (Cegeka's Agile Software Factory) document was used. In this document, a detailed explanation on the agile way of working in Cegeka is provided. The fact that the methodology described in the document was not verified by the actual implementers of it, raises a doubt regarding the assumption that all steps are followed in the distributed development process.

In total, there are three main limitations of this research. Firstly, the time boundaries as imposed by the context in which the research was developed and, secondly, the restricted amount of detailed information concerning the actual communication between the Business Analyst and the Scrum Master, which increased the level of our assumptions and the doubt for those assumptions because of the inability of communicating with the Business Analyst and the Scrum Master.

CONCLUSIONS

We began this research with one main question:

> *"How to improve the requirements communication between the Business Analyst and the Scrum Master so that the developers have a better understanding of the company and product vision?"*

For understanding and answering the problem better we subdivided this issue in three parts: a) How to incorporate company and product vision into the software requirements or through the whole agile process?; b) How to make communication between Business Analyst and Scrum Master better?; c) How to make developers more aware of the company and product vision?

Answers to these issues are based on both interviews with Mr. Gerard Murre (Director of the Shared Software Factory-Netherlands) and Mr. Laurentiu Oprea (Business Unit Manager-Romania) and on academic research. It appears that communication is not a simple problem in companies. Indeed the good comprehension of requirements depends on the person who gives the requirements (the stakeholders) and the one who receives them (the developers).

Recommendations

Requirements organization needs to be sensitive to how people collaborate or are influenced by their way of working. Of course communication can be improved with web-based user story tools or good documentation which will help to incorporate vision and strategy such as roadmaps. Emphasis on the coordination of the communication can also enhance the collaboration of the distributed team. Additionally, applying scrum sprints for the requirements refinery can address the issues of the requirements clarification.

Due to the limited information we have on how the Scrum Master and the Business Analyst from Cegeka communicate, we cannnot give a very specific answer but just a general solution and best practices. Nonetheless it should be a good starting point for improving requirements communication within the distributed software development process between the Romanian and the Dutch sites of Cegeka.

REFERENCES

Boehm, B. W. (1984). Verifying and Validating Software Requirments and Design Specifications. *IEEE*, 75-88.

Burg, J. F. (1997). *Linguistic Instruments in Requirements Engineering*. Amsterdam: IOS Press.

Cao, L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*, 61-67.

Carmel, E., & Agarwal, R. (2001). Tactical approaches for alleviating distance in global software development. *Software, IEEE, 18(2)*, 22-29.

Cegeka vision and mission: ICT in close cooperation. (n.d.). Retrieved from Cegeka - In close cooperation: http://www.cegeka.be/EN/Information/AboutCegeka/Visionmission

Cegeka's Agile Software Factory. (n.d.). *Agile @ Cegeka.* Retrieved from Cegeka - In close cooperation: http://www.cegeka.be/portals/118/AgileCegeka_EN.pdf

Christel, M. G., & Kang, C. K. (1992). *Issues in Requirements Elicitation.* Pennslyvania: CMU Press.

Cohn, M. (2004). *User Stories Applied for Agile Software Development.* Pearson Education.

Cornford, T., & Smithson, S. (2006). *Project research in information systems: a student's guide.* (Second ed.). New York, USA: Macmillan, Palgrave.

Dorairaj, Siva., Noble, James.,& Malik, Petra (2011). Effective Communication in Distributed Agile Software Development Teams. XP 2011 LNBIP 77 (pp 102 – 116). Springer-Verlag Berlin Heidelberg .

Fenton, N. E. (1991). *Software Metric: A Rigorous Approach.* London: Chapman & Hall.

Fogelström, N. D., Gorschek, T., Svahnberg, M., & Olsson, P. (2010). The impact of agile principles on market- driven software product development. *Journal of Software Maintenance and Evolution: Research and Practice, 22(1)*, 53-80.

Goguen, J., & Jirotka, M. (2004). *Requirements Engineering: Social and Technical lssues.* London: Academic Press.

Goguen, J., & Linde, C. (2003). Techniques for Requirements Elicitation. *1st IEEE International Symposium on Requirements Engineering*, (pp. 152-164). San Diego.

Gottesdiener, E. (2009). *Agile Requirements by collaboration.*

Gottesdiener, E. (2011). *Agile Requirements: Not an Oxymore.*

Hashmi, S. I., Ishikawa, F., & Richardson, I. (2013). A Communication Process for Global Requirements Engineering. *Proceedings of the 2013 International Conference on Software and System Process* (pp. 136-140). San Francisco, USA: ACM.

Herbsleb, J. D., & Mockus, A. (2003). An empirical study of speed and communication in globally distributed software development. *Software Engineering, IEEE Transactions, 29(6)*, 481-494.

Holmström, H., Fitzgerald, B., Ågerfalk, P. J., & Conchúir, E. Ó. (2006). Agile practices reduce distance in global software development. *Information Systems Management, 23(3)*, 7-18.

Holtzblatt, K., & Beyer, H. R. (2005). Requirements Gathering: The Human Factor. *Communications of the ACM*, 31- 32.

Hossain, E. (2008). Coordinating mechanisms for Agile Global Software Development. *IEEE International Conference on Global Software Engineering (ICGSE 2008)* (pp.257-263). IEEE.

ICT Outsourcing Services. (n.d.). Retrieved 2013, from Cegeka - In close cooperation: http://www.cegeka.be/EN/Productsandservices/Outsourcing/Workplace

Kotonya, G., & Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques.* Wiley.

Layman, L., Williams, L., Damian, D., & Bures, H. (2006). Essential communication practices for Extreme Programming in a global software development team. *Information and Software Technology, 48(9)*, 781-794.

Layton, M. C. (2012). *Agile Project Management for Dummies.*

Lehman, M. M. (1980). Programs, Life Cycles, and Laws of Software Evolution. *Proceedings of IEEE*, 1060-1076.

Liskin, O., & Schneider, K. (2012). Improving Project Communication with Virtual Team Boards. *IEEE Seventh International Conference on Global Software Engineering Workshops*, (pp.35-36). Rio Grande do Sul, Brazil.

Morgan, R., & Maurer, F. (2006). MasePlanner: A Card-Based Distributed Planning Tool for Agile Teams. *IEEE International Conference on Global Software Engineering (ICGSE'06)* (pp. 1-5). Florianopolis, Brazil: IEEE Computer Society.

Morrison, D. (2009). *Agile Product Management Framework*. Retrieved from All about

Product Management: http://allaboutproductmanagement.blogspot.nl/2009/03/agile-product-management-framework.html

Mylopoulos, J., Chung, L., & Yu, E. (1999). From Object-Oriented to Goal-Oriented Requirement Analysis. *Communications of the ACM, 42*(1), 31-37.

Nuseibeh, B., & Easterbrook, S. (2000). *Requirements Engineering: A Roadmap.* Limerick: ACM Press.

Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements Engineering and Agile Software Development. *IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* (pp. 1-6). New Jersey: IEEE.

Pichler, R. (2013). *Agile Product Planning: Vision, Strategy, and Tactics*. Retrieved from http://www.romanpichler.com/blog/product-planning/agile-product-planning-vision-strategy-tactics/

Posner, M. I. (1993). *Foundations of Cognitive Science.* Massachusettes: MIT Press.

Pragmatic Marketing & Enthyosis. (2012). *Living in a Agile World:The Strategic Role of Product Management WheN Development Goes Agile.*

Qumer, A., & Henderson-Sellers, B. (2008). A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software, 81(11)*, 1899-1919.

Ramesh, B., Cao, L., Mohan, K., & Xu, P. (2006). Can distributed software development be agile? *Communications of the ACM, 49(10)*, 41-46.

Rautiainen, K., Lassenius, C., & Sulonen, R. (2002). 4CC: A Framework for Managing Software Product Development. *Engineering Management Journal, 14(2)*, 27-32.

Rees, M. J. (2002). A Feasible User Story Tool for Agile Software Development? *Proceedings of the Ninth Asia-Pacific Software Engineering Conference (APSEC'02).* Queensland, Australia: IEEE Computer Society.

Royce, W. (2009). Improving Software Economics. *IBM Corporation Software Group*, 1-40.

Sharp, H., Robinson, H., & Petre, M. (2009). The role of physical artefacts in agile software development: Two complementary perspectives. *Interacting with Computers, 21*, 108- 116.

Singh, M. (2008). U-Scrum: An Agile Methodologyfor Promoting Usability. *Agile, 2008. AGILE'08. Conference* (pp. 555-560). IEEE.

Sommerville, I. (2007). *Software Engineering* (8 ed.). Pearson Education.

Thomson, B. (2012). *Creating a Strategic Product Plan*. Retrieved from Pragmatic Marketing - Practical Training. Proven Results.: http://www.pragmaticmarketing.com/resources/creating-a-strategic-product- plan

Turk, D., France, R., & Rumpe, B. (2002). Limitations of Agile Software Processes. *Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP 2002)*, (pp. 43-46).

Vähäniitty, J., & Rautiainen, K. T. (2008). Towards a Conceptual Framework and Tool Support for Linking Long-term Product and Business Planning. *Proceedings of the 1st International workshop on Software development governance* (pp. 25-28). ACM.

Vlaanderen, K., Jansen, S., Brinkkemper, S., Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software products management. *Information and Software Technology, 53(1)*, 58-70.

Wiegers, K. E. (2009). *Software Requirements* (2 ed.). O'Reilly.