LOUVAIN
School of Management
RESEARCH INSTITUTE

WORKING PAPER

2016/25

Extending I-Tropos for Agile Software Development

Hassan Haidar,
Manuel Kolp,
Louvain School of Management Research Institute

Duc Dung Do,
Louvain School of Management

Yves Wautelet,
KULeuven

LOUVAIN SCHOOL OF MANAGEMENT RESEARCH INSTITUTE

Louvain School of Management Research Institute
Working Paper Series
Editor: Prof. Frank Janssen
(president-ilsm@uclouvain.be)

Extending I-Tropos for Agile Sotware Development

Hassan Haidar, Louvain School of Management Research Institute
Manuel Kolp, Louvain School of Management Research Institute
Duc Dung Do, Louvain School of Management
Yves Wautelet, KULeuven

Summary

Combining an iterative development process such as I-Tropos with an agile framework such as Scrum helps software professionals to deal with the social aspects and human needs. On the one hand, Iterative-Tropos (I-Tropos) is a "development process using coarse grained (i.e. high-level) and social-oriented requirements models, to drive the software development both in terms of project management and forward engineering (transformational) techniques". On the other hand, Scrum is an agile framework for managing the development of complex products and services, especially software development. It is applicable to any project with aggressive deadlines, complex requirements and a degree of uniqueness.

The objective of this working paper is to propose an integration of Scrum in I-Tropos in order to create a new hybrid methodology called I-Tropos-Extended. In fact, the integration of these two approaches aims to empower both the benefits of I-Tropos and Scrum. Consequently, the proposed approach could provide several benefits such as efficient software project management, continuous organizational modeling and requirements acquisition, early implementation, continuous testing and modularity, etc.

Keywords : I-Tropos, Scrum, Unified Process, Agile Software Development, Project Management.

JEL Classification: L86

Corresponding author:
Hassan Haidar
CEMIS
Louvain School of Management Research Institute / Campus of Louvain-la-Neuve
Université catholique de Louvain
Place des Doyens 1
B-1348 Louvain-la-Neuve, BELGIUM
Email : hassan.haidar@uclouvain.be

1. Introduction

Software systems and applications influence people's lives every day. So, the massive use of information systems (IS) in most of the crucial aspects of everyday life – i.e. software systems developed to support human activities and to cope with social problems – makes it more evident to the professionals that the software developers have to deal with social aspects and human needs [1]. In the literature, many approaches were presented to handle the job. Among these approaches, the most outstanding one is the iterative development approach. In fact, iterative development consists in the development of a series of releases having different scopes and increasing completeness. It allows systematically collecting users' opinions and desiderata on the software under development and targeting more complex and socio-technical systems. Therefore, professionals, notably through the application of the *Unified Process* [2] [3] [4] and the agile initiative, are increasingly using iterative development.

On the one hand, *Iterative-Tropos (I-Tropos)* is a "development process using coarse grained (i.e. high-level) and social-oriented requirement models, to drive the software development both in terms of project management and forward engineering (transformational) techniques" [1]. The methodology aims to fill the iterative life cycle gap of Tropos [5] through a phased template adapted from the UP and custom project management framework driven by high-level entities allowing to model system requirements [6]. On the other hand, *Scrum* is an agile framework for managing the development of complex products and services [7], especially software development. Scrum is a project management framework that is applicable to any project with aggressive deadlines, complex requirements and a degree of uniqueness. In Scrum, projects move forward via a series of iterations called sprints. Each sprint is typically two to four weeks long [8].

According to some researchers and software development practitioners, a hybrid approach can help optimize the software development lifecycle by combining several methodologies. *OpenUP* [9] is such a successful methodology that relates to both RUP (Rational Unified Process) and *agile software development* [10].

The objective of this working paper is to propose an integration of Scrum in I-Tropos in order to create a new hybrid methodology called I-Tropos-Extended. In fact, the integration of these two approaches aims to empower both the benefits of I-Tropos and Scrum. Consequently, the proposed approach could provide several benefits such as efficient software project management, continuous organizational modeling and requirements acquisition, early implementation, continuous testing and modularity, etc.

The working paper is organized as follows: *Section 2* introduces an overview of the Scrum methodology. *Section 3* presents a description of I-Tropos. *Section 4* proposes a new methodology named *I-Tropos-Extended* which combines I-Tropos and Scrum. A case study will be presented in the *Section 5* to illustrate the Extended I-Tropos. Finally, *Section 6* concludes the paper and points out limitations and further work.

2. Scrum

Scrum is an agile management process that uses an incremental and iterative approach to help manage and organize the software development process. A Scrum is a project management framework that is applicable to any project with aggressive deadlines, complex requirements and a degree of uniqueness. In Scrum, projects move forward via a series of iterations called sprints. Each sprint is typically two to four weeks long.

According to [7], a sprint is defined as a set period of time during which specific work has to be completed and made ready for review. Sprints are iterations or cycles of up to a calendar month. At the end of each sprint, the performed work should create something of tangible value to the stakeholders. Thus, sprints are time-boxed so they always have a fixed start and end dates, and generally, they should all be of the same duration.

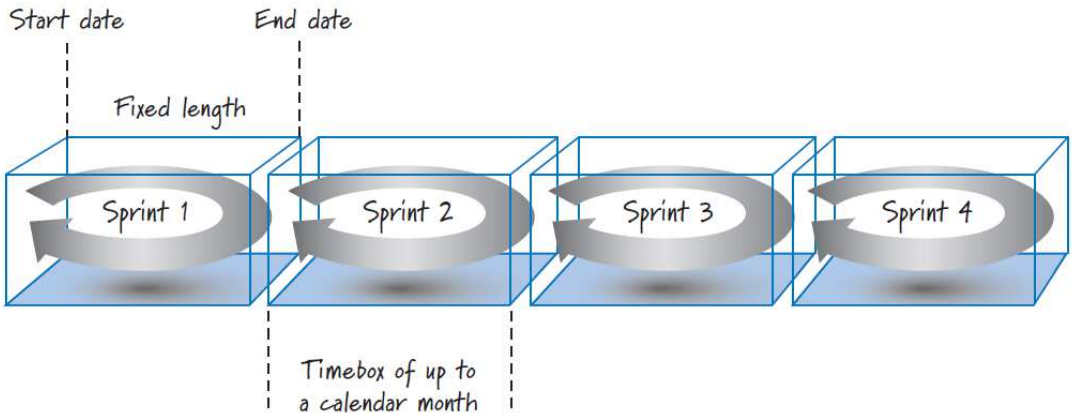

Fig. 1 Sprint characteristics [7]

2.1 Common Concepts in Scrum

Scrum Team: A typical scrum team counts between five and nine people, but Scrum projects can easily scale into the hundreds. However, Scrum can easily be used by one-person teams and often is. This team does not include any of the traditional software engineering roles such as programmer, designer, tester or architect. Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint. Scrum teams develop a deep form of camaraderie and a feeling that “we’re all in this together.”

Product Owner: The product owner is the project’s key stakeholder and represents users, customers and others in the process. The product owner is often someone from product management or marketing, a key stakeholder or a key user.

Scrum Master: The Scrum Master is responsible for making sure that the team is as productive as possible. He does this by helping the team use the Scrum process, by removing impediments to progress, by protecting the team from outside, and so on.

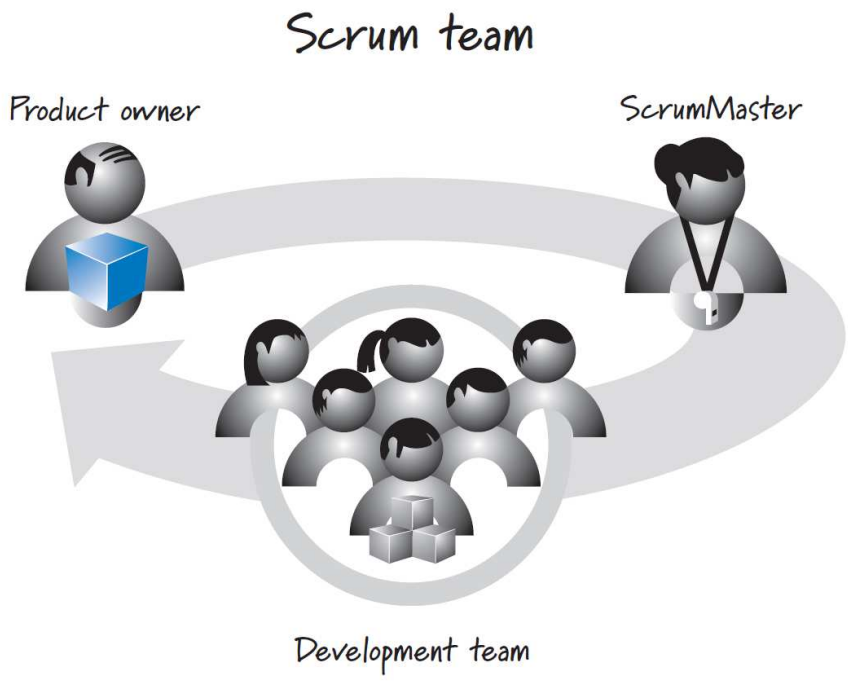

Fig. 2 Scrum roles [7]

Product backlog: The product backlog is a prioritized feature list containing every desired feature or change to the product. Note: The term "backlog" can get confusing because it is used for two different things. To clarify, the product backlog is a list of desired features for the product. The sprint backlog is a list of tasks to be completed in a sprint.

Sprint planning meeting: At the start of each sprint, a planning meeting is held, during which the product owner presents the top items on the product backlog to the team. The Scrum team selects the work they can complete during the coming sprint. That work is then moved from the product backlog to a sprint backlog, which is the list of tasks needed to complete the product backlog items the team has committed to complete in the sprint.

Daily Scrum: Each day during the sprint, a brief meeting called the daily scrum is conducted. This meeting helps set the context for each day's work and helps the team stay on track. All team members are required to attend the daily scrum.

Sprint review meeting: At the end of each sprint, the team demonstrates the completed functionality at a sprint review meeting, during which, the team shows what they accomplished during the sprint. Typically, this takes the form of a demonstration of the new features, but in an informal way; for example, PowerPoint slides are not allowed. The meeting must not become a task in itself, nor a distraction from the process.

Sprint retrospective: Also at the end of each sprint, the team conducts a sprint retrospective, which is a meeting during which the team (including its Scrum Master and product owner) reflects on how well Scrum is working for them and what changes they may wish to make for it to work even better.

2.2 Scrum framework

Fig. 3 Scrum Framework [7]

To understand the Scrum framework, [7] explains that one should start on the left side of Figure 3 and work clockwise around the principal looping arrow (the sprint).

First, the ***product owner*** has a vision of what he wants to create (the big cube). Because the cube can be large, through an activity called grooming, it is broken down into a set of features that are collected into a prioritized list called the ***product backlog***.

Then, a sprint starts with ***sprint planning***, encompasses the development work during the sprint (called sprint execution), and ends with the review and retrospective. The large, looping arrow that dominates the center of the figure 3 represents ***the sprint***. The number of items in the product backlog is likely to be more than a development team can complete in a short-duration sprint. For that reason, at the beginning of each sprint, the development team must determine a subset of the product backlog items they believe they can complete—an activity called ***sprint planning***, shown just to the right of the large product backlog cube.

To acquire confidence that the ***development team*** has made a reasonable commitment, the ***team members*** create a second backlog during sprint planning, called the ***sprint backlog***. The sprint backlog describes, through a set of detailed tasks, how the team plans to design, build, integrate, and test the selected subset of features from the product backlog during that particular sprint.

Next is ***sprint execution***, where the development team performs the ***tasks*** necessary to realize the selected ***features***. Each day during sprint execution, the team members help manage the flow of work by conducting a synchronization, inspection, and adaptive planning activity known as the ***daily scrum***. At the end of sprint execution, the team has produced a ***potentially shippable product increment*** that represents some, but not all, of the product owner's vision.

The Scrum team completes the sprint by performing two ***inspect-and-adapt*** activities. In the first, called the ***sprint review***, the stakeholders and Scrum team inspect the product being built.

In the second, called the ***sprint retrospective***, Scrum team inspects the Scrum process being used to create the product. The outcome of these activities might be adaptations that will make their way into the product backlog or be included as part of the team's development process.

At this point, the Scrum sprint cycle starts over, beginning anew with the development team determining the next most important set of product backlog items they can complete. After an appropriate number of sprints have been completed, the product owner's vision will be realized and the solution can be released.

2.3 Scrum Process

Scrum is based on iterative and incremental development as shown in Figure 4.

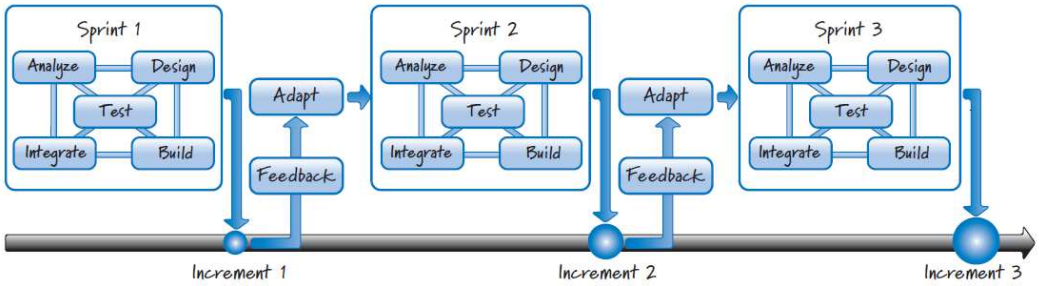

Fig. 4 Scrum Iterative and incremental Process [7]

Iterative development acknowledges that we will probably get things wrong before we get them right and that we will do things poorly before we do them well [11]. In fact, the team uses multiple passes to improve what we are building so that we can converge on a good solution. Thus, iterative development is an excellent way to improve the product as it is being developed.

Incremental development is based on the age-old principle of "Build some of it before you build all of it". Incremental development gives team members important information that allows them to adapt their development effort and change how they proceed. The biggest drawback to incremental development is that by building in pieces, they risk missing the big picture [7].

Consequently, scrum leverages the benefits of both iterative and incremental development, while negating the disadvantages of using them individually. Scrum does this by using both ideas in an adaptive series of sprints (see Figure 4).

Moreover, at the heart of Scrum are the principles of *inspection*, *adaptation*, and *transparency*. In Scrum, team members inspect and adapt not only what they are building, but also how they are building it (see Figure 5).

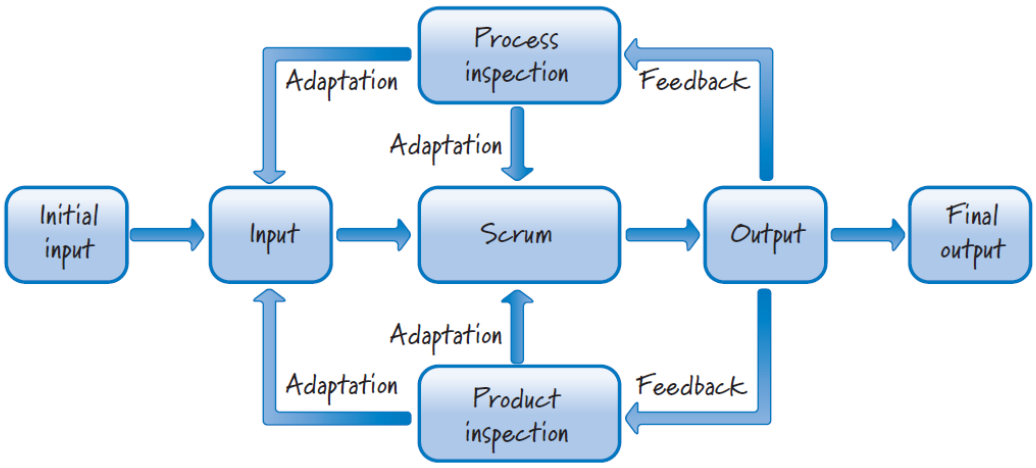

Fig. 5 Scrum Process Model [7]

2.4 Main benefits of Scrum

Organizations that apply diligently Scrum experience different benefits. [7] mentions some of these benefits as seen in Figure 6:

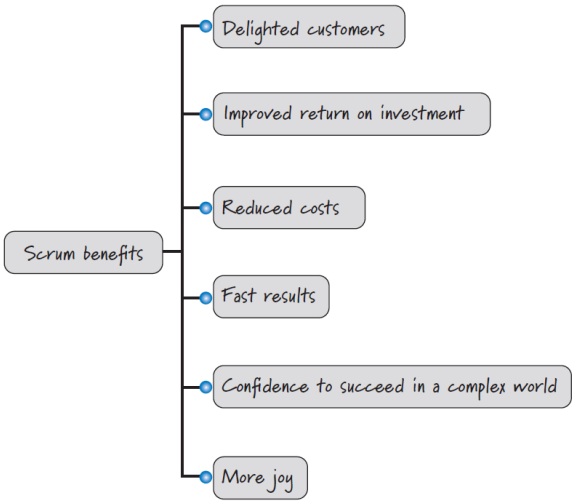

Fig. 6 Scrum benefits [7]

These organizations are repeatedly delighting their customers by giving them what they really want, not just the features they might have specified on the first day when they knew the least about their true needs. They are also seeing an improved return on investment by delivering smaller, more frequent releases. And, by relentlessly exposing organizational dysfunction and waste, these organizations are able to reduce costs.

Furthermore, Scrum focuses on delivering working, integrated, tested, business-valuable features. Each iteration leads to results being delivered fast. Scrum is also well suited to help organizations succeed in a complex world where they must quickly adapt based on the interconnected actions of competitors, customers, users, regulatory bodies, and other stakeholders.

3. I-Tropos

Referring to [13], the I-Tropos process was built-up based on previously validated methodologies that have been combined. Indeed, I-Tropos basically constitutes an enhanced version of the Tropos [12] process where extensions are intended to furnish techniques and tools (driven by services) to deal with scalability and develop iteratively (see also Figure 1). It has thus been built on the basis of:

- The methodological foundations of the Tropos process allowing defining a series of disciplines wherein Tropos' activities are grouped. This represents the vertical dimension of the process;
- The iterative perspective of the UP. Indeed, the UP proposes a series of phases, which are used and revised here within the context of Tropos. This represents the horizontal dimension of the process.

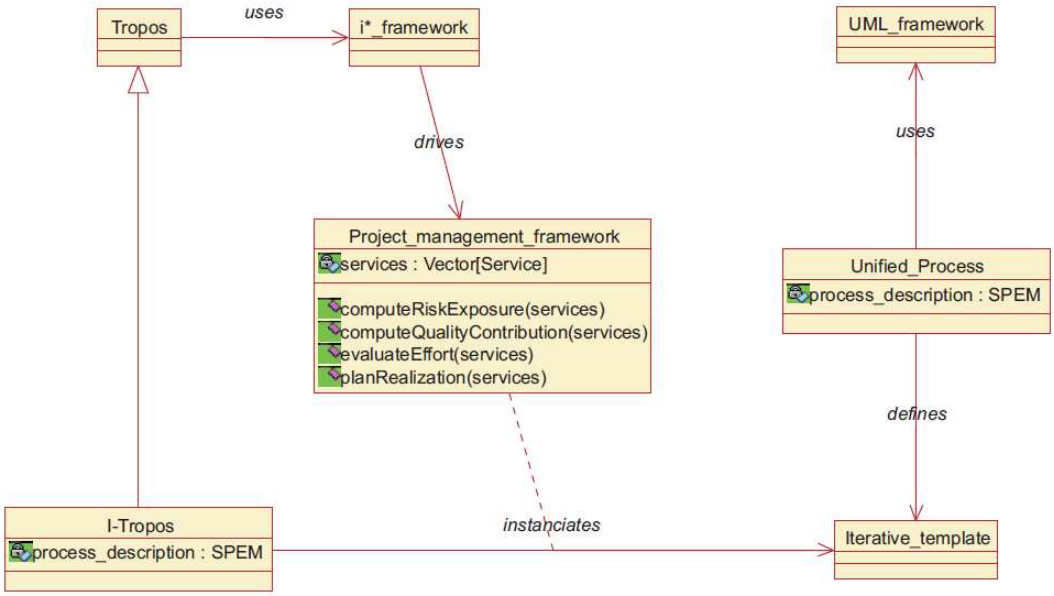

Fig. 7 The project management framework within development methods [13]

3.1 Core disciplines of I-Tropos

Organizational Modeling, concerned with the understanding of a problem by studying an organizational setting.

Requirements Engineering, where the system-to-be is described within its operational environment, along with relevant functions and qualities.

Architectural Design, where the system's global architecture is defined in terms of subsystems interconnected through data, control and other dependencies.

Detailed Design, where the behavior of each architectural component is further refined.

Implementation, where an executable release of the application based on the detailed design specification is produced.

Test, where the quality of the executable release is evaluated.

Deployment, where the test of the software in its final operational environment takes place.

Software Project Management, where management is based on a project management framework driven by high-level entities that allow modeling system requirements. This PM framework covers risk, quality and time management aspects at the highest decision level, taking Threats and Quality Expectations (QE) evaluation directly into account in (iterative) development planning; it is positioned with respect to Tropos and the UP in *Figure 7*.

3.2 Process phases to be planned within I-Tropos

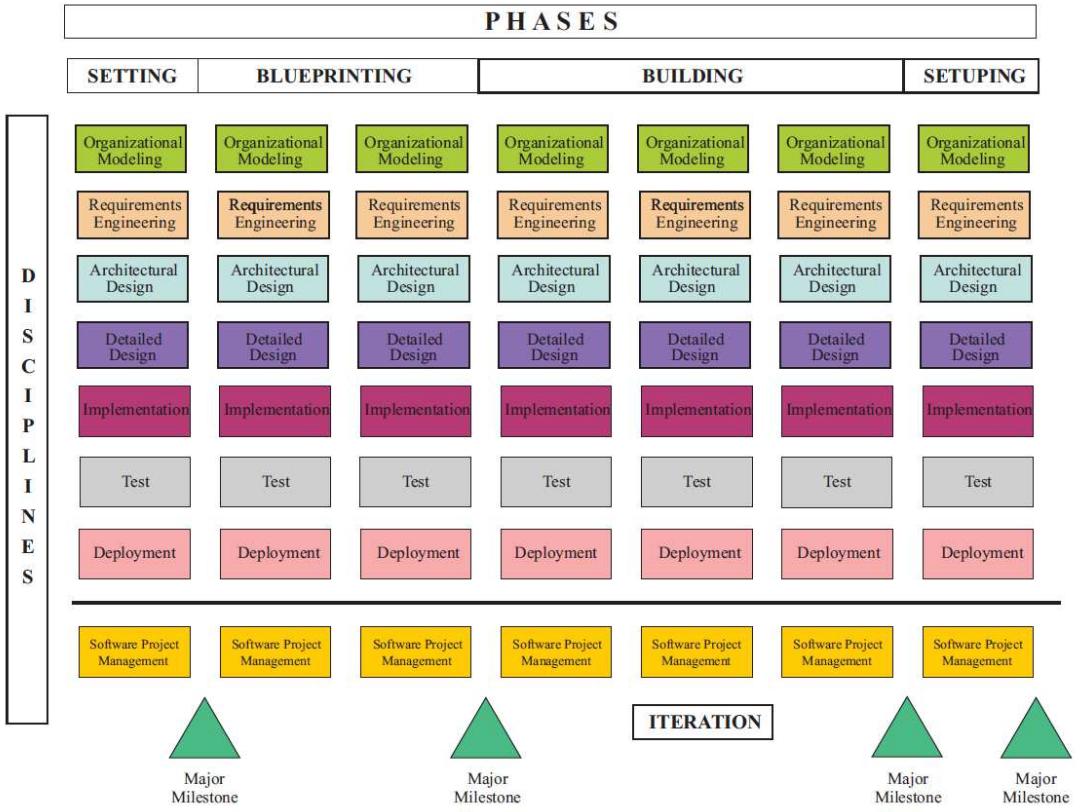

Fig. 8 I-Tropos: Iterative Perspective [13]

[13] presents the I-Tropos process. In fact, I-Tropos' phases are made upon a refinement of the UP phases as illustrated in *Figure 8* (milestones expressed hereafter are based on metrics expressed in the UP, see [4]); each one is made of one or more iterations. Disciplines have gone through sequentially; as stressed before major milestones separate the phases. Each of them has its own goal:

The ***Setting phase*** is designed to identify and specify most of stakeholders' requirements, have a first approach of the scope's environment, identify and evaluate project's threats and identify and evaluate QE;

The ***Blueprinting phase*** is designed to produce a consistent architecture for the system on the basis of the identified requirements, eliminate most risky features in priority and evaluate first blueprints/prototypes to stakeholders;

The ***Building phase*** is designed to build and evaluate each aspect of a working application and validate developments;

The ***Setuping phase*** is designed to finalize production, train users and document the system.

Each phase is made of several iterations and it is up to the development team to determine the exact amount of iterations they want to include in the Blueprinting, Building phases and Setuping phases (typically the Setting phase is a unique iteration). Thus, an "I-Tropos development" is made of disciplines repeated iteratively while the effort spent on each discipline varies from one iteration to another.

3.3 I-Tropos roles

Based on Software Process Engineering Meta-model (SPEM) notations, [14] implements the I-Tropos process roles. The diagram below (see Figure 9) shows the different roles implied in the I-Tropos process as well as their hierarchy.

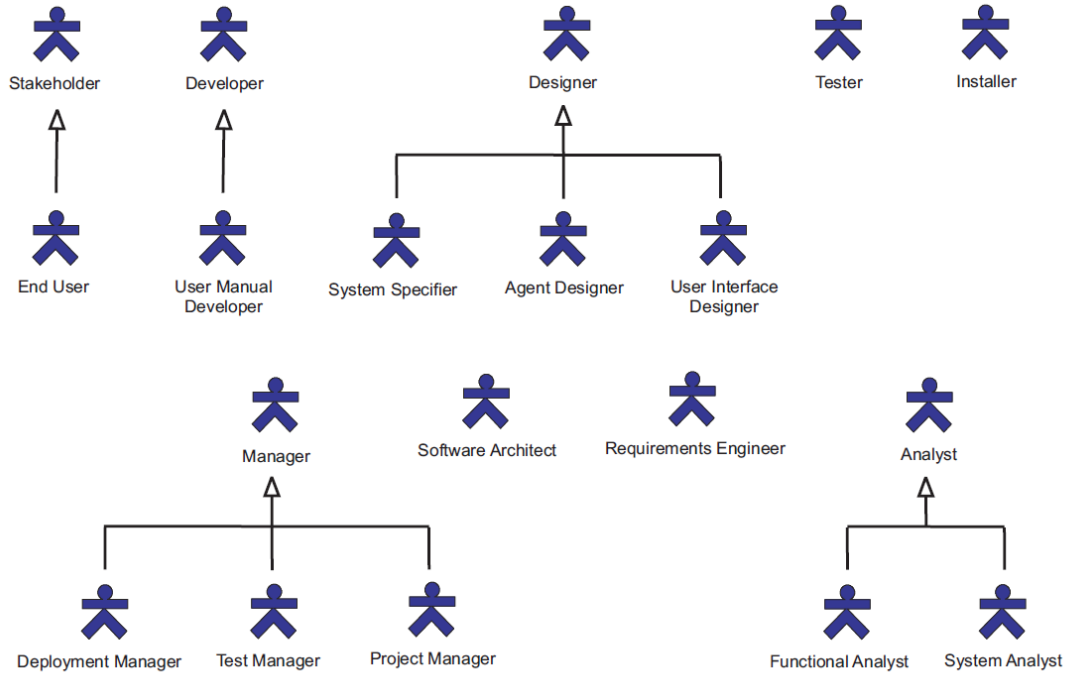

Fig. 9 I-Tropos process roles [13]

4. Extending I-Tropos by combining Scrum

In order to boost organization's benefits, this working paper presents a methodological approach (i.e. a way to deal with an iterative life cycle in a (huge) software project requiring a high business and IT alignment) that extends I-Tropos by integrating Scrum process, principals and framework. This section proposes an agile version of I-Tropos called I-Tropos-Extended. This methodology has the essential characteristics of I-Tropos that applies iterative and incremental approaches with Scrum.

4.1 Suggested roles

In the extending approach, the following roles are suggested to be undertaken:

Stakeholder represents interest groups whose needs must be satisfied by the project. This role may be played by anyone who is (or potentially will be) materially affected by the outcome of the project.

Analyst represents the customer and end-user concerns by gathering input from stakeholders to understand the problem to be solved and by capturing and setting priorities for requirements.

Architect is in charge of designing the software architecture, which includes making the key technical decisions that constrain the overall design and implementation of the project.

Developer is in charge of developing a part of the system, including designing it to fit into the architecture, and then implementing, unit-testing, and integrating the components that are part of the solution.

Deplorer is in charge of all of the activities that make a software system available for use.

Tester is in charge of the core activities of the test effort, such as identifying, defining, implementing, and conducting the necessary tests, as well as logging the outcomes of the testing and analyzing the results.

Project Manager leads the planning of the project in collaboration with stakeholders and team, coordinates interactions with the stakeholders, and keeps the project team focused on meeting the project objectives.

Any Role represents anyone on the team that can perform general tasks.

4.2 Disciplines

Requirements include I-Tropos' Organizational Modeling and Requirements Engineering disciplines. For example, i*, UML, BPMN, etc.

Design corresponds to I-Tropos' Architectural Design and Detailed Design disciplines.

Development combines with I-Tropos' implementation and Test disciplines.

Deployment corresponds to I-Tropos Deployment discipline.

4.3 I-Tropos-Extended process

I-Tropos-Extended uses exactly the same process phases as I-Tropos. However, the Scrum methodology will be integrated into each phase to organize the software development process and improve the management of the day-to-day activities. The phases of the proposed process are:

Setting: identify and specify most stakeholders' requirements, have a first approach of the scope's ecosystem, identify and evaluate project's threats and identify and evaluate quality factors.

Blueprinting: produce a consistent architecture for the system on the basis of the identified requirements, eliminate most risky features in priority and evaluate blueprints/prototypes to stakeholders.

Building: build a working application and validate achieved developments.

Setuping: finalize production, train users and document the system.

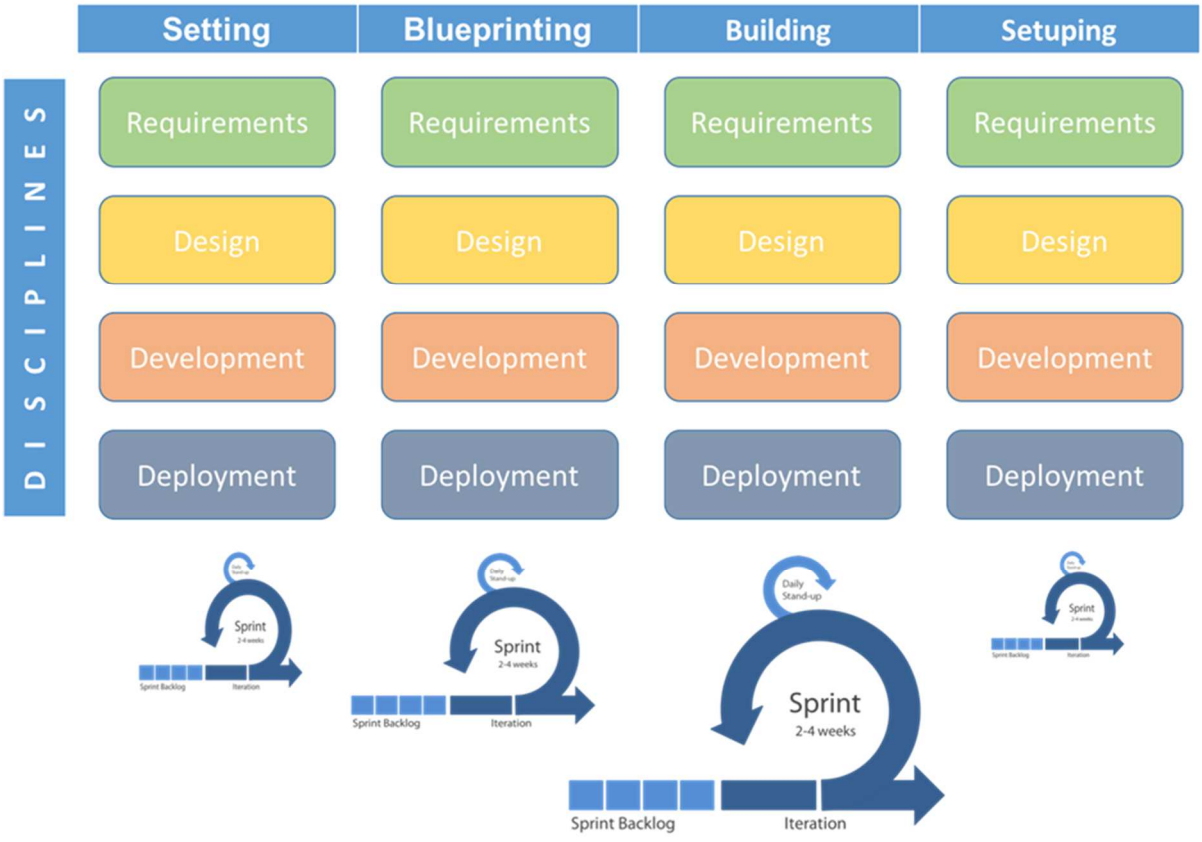

Fig. 10 I-Tropos-Extended iterative perspective

4.4 Discussion

The goal for Setting and the Blueprinting phases is to learn about the organization and the environment where the application will operate, understand the user's needs and expectations, gather requirements, define the project scope, have an initial risk assessment, and establish an initial baseline for the software architecture.

The main goal of the ***Setting*** phase is to have an agreement among stakeholders about what were the objectives of the project such as what would be or would not be included in the product. For this reason, it is important to understand the structure and dynamics of the organization as well as to learn how the application will be used and what the user's expectations are to this. Therefore, the outputs produced during the setting phase are mainly analysis disciplines (requirements) such as the business vision, the business use-case model, the domain model, a glossary, the product vision, an initial use case model, a software development plan, and the initial risk list. The Scrum Team of the Setting phase includes Business Analyst who is the Scrum master, Stakeholder, Project Manager and potentially Architect. *1-2 sprints* should be enough for this phase.

One of the main goals during the ***Blueprinting*** phase is to finish up the elaboration of the use cases and to have a better understanding of the most critical use cases that would drive the software architecture. This is important so that the team could elaborate an initial baseline of the architecture. For this purpose, the effort is spent mainly on Design discipline. At the end of this phase, the development team should have a clear and detailed understanding of the user's requirements. In this way, the development team could validate and baseline the architecture of the system and could refine the development plan to reflect more precisely on what needed to be done until the delivery of the project. It is essential to get feedback from the stakeholders to ensure the success of the project. In addition, it is necessary to create the product backlog with the list of features and prioritize them. The Scrum master during the Blueprinting phase should be an architect. The Scrum Team also includes a Project Manager, a Functional Analyst, and a Technical Analyst. *1-3 sprints* could be used to accomplish the objective of the Blueprinting phase.

The aim of the two phases ***Building and Setuping*** is to implement and to deliver a product which matches perfectly the customer's requirements and expectations. To ensure that, it was important to have an executable architecture so that the team could concentrate on implementing the rest of the functionality and performing the necessary tasks to deliver the project. To accomplish these tasks, the Scrum practices are integrated into the process so that the team would plan, organize, and collaborate to reach a tangible goal. The Scrum master during these two phases should be Project Manager. Another choice for the Scrum Master could be Technical Analyst for the Building phase and Deplorer for the Setuping phase. The Scrum Team for the Building phase includes Technical Analyst, Developer, Project Manager and Any Role. As for the Setuping phase, the Scrum team includes Deplorer, Project Manager and potentially Technical Analyst in case of a deployment issue. *3-8 sprints* for the Building phases and one sprint for the Setuping phase should be enough for a medium-size project.

5. Case Study

This section describes the Mainsys Engineering project that will be used as a case study for this working paper.

Mainsys Engineering (ME) SA, an IT services and software company specialized in the financial sector located in Brussels, is developing a new module "***Client subscription request***". The aim is to provide BuyWay Personal Finance SA's new clients with possibility of making an online subscription. The internet users can thus use a web interface to send their subscription request and create a new bank account.

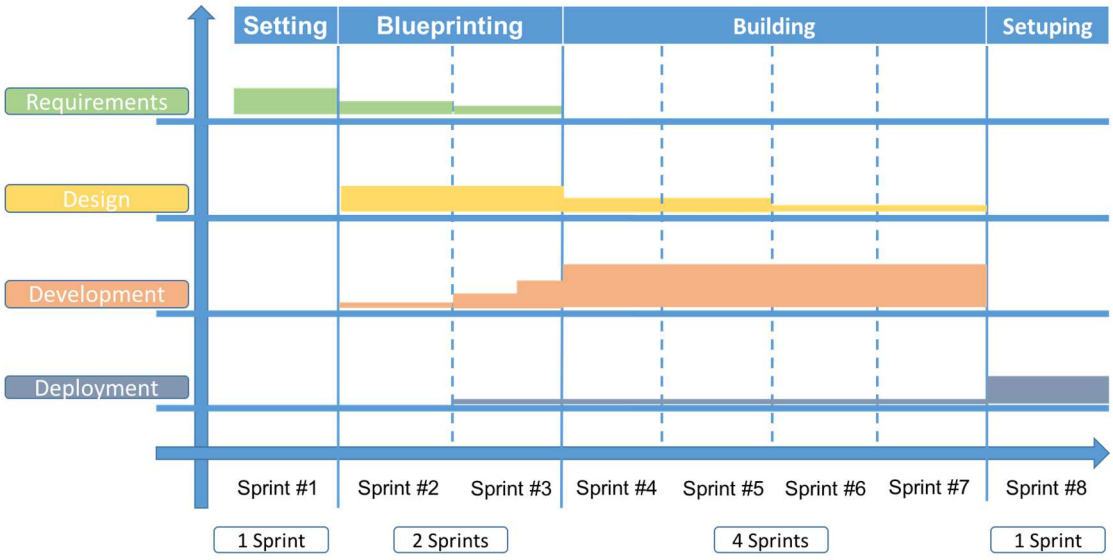

Fig. 11 I-Tropos-Extended profile for the "Client subscription request" project

The completely I-Tropos-Extended project profile is shown in Figure 11 above. Rectangles represent the relative effort spent on each of the disciplines during each of the phases. The effort spent on requirement discipline is higher into the Setting and marginal for the other ones. The Design is higher in the Blueprinting and marginal for the Building phase. On the contrary, the Development one is marginal for the Blueprinting phase and higher in the Building phase. Almost all the effort spent on deployment lies in the Setuping phase.

The "Client subscription request" project is divided into *8 sprints*: 1 for the Setting phase, 2 for the Blueprinting, 4 for the Building and finally 1 sprint for the Setuping.

The main goal of the *sprint#1* is to have an agreement among *Mainsys* and *Buy Way* about what are the objectives of the project such as what will be included in the client request form. The Scrum Team for this sprint includes a Business Analyst of Mainsys who is the Scrum master, a Stakeholder who represent Buy Way and the Project Manager.

The goal for the *sprint#2 and #3* of the Blueprinting phase is to validate and baseline the architecture and create a more fine-grained development plan, to prepare the development environment, process and tools. The Scrum master during these two sprints will be an Architect. The attendees to the Scrum meeting will be Project Sponsor, Functional Analyst, Technical Analyst. During the *sprint#3*, the development team could participate in the scrum meeting. At this meeting, the development team has

the opportunity to demonstrate the user-interface prototype to the stakeholders. The demonstration of the user-interface prototype elicited feedback that included the addition of some nice-to-have features, some small changes to improve the software. During the scrum meeting, the project sponsor who will play the role of the product owner, helps identify the features that have more priority. This helps the development team prioritize the list of features that will form the product backlog. After the product backlog was prioritized, the development team begins to estimate the effort needed to implement each feature in the product backlog. The development team then elaborates a development plan based on the product backlog and other activities needed to accomplish the project. According to the development plan, the Building phase is organized to have four sprints and two releases. The first release is planned at the end of the *sprint#5* and the second release is planned at the end of the *sprint#7*. Most of the activities from *sprint#4 to sprint#7* will be focused on implementing the main functionality of the system and any new requests.

Sprint#4: At the beginning of the sprint, the development team revises the software development plan and the product backlog to select the features that are going to form the sprint backlog. The development team is needed to set up the development environment for the application.

Sprint#5: The goal for this sprint was to deliver the first release. Thus, one of the first activities carried out during this sprint is to add more detail to the development plan about the tasks needed to be carried out in this sprint. In addition, the plan for the next sprint will be updated.

First Release: The purpose of the first release is to show the stakeholders the advances the development team has achieved so far. This first release will provide an opportunity for the stakeholders to try and test the application and provide new feedback. With this purpose, the first release will be deployed to an area of the development server so that stakeholders could easily access it.

Sprint#6: The development team continues to implement the following features from the product backlog while awaiting the stakeholders' feedback. Some test scenarios will be elaborated to map functionality of the application. All bugs found will be added as tasks to be corrected in the next sprint.

Sprint#7: At the beginning of this sprint, the development team together with the Project Manager, who plays the role of the Scrum Master, takes a decision based on the features left in the product backlog. The decision is that the development team is going to be able to deliver a functional system that includes all the core requirements as part of the second release. For this sprint, the development team dedicates all their efforts to complete the core requirements and deliver a version of the product.

Sprint#8: After the second release, the project is in the last phase of Building and enters the Setuping phase. In the past sprint, the development team had released an operational product that fulfilled the core requirements. The main functionality is in place and all bugs are corrected. The software is ready for the deployment and the final product obtained the approval from the project sponsor.

6. Conclusion

The proposed project management framework in this working paper is tailored to the combination of I-Tropos with Scrum. The first threat to the validity of the proposed approach is the need to perform deep studies on the different transformation techniques (classical UML at design stage, i* mappings, and other development languages) in order to stabilize the application of the approach.

Furthermore, this extended approach needs to be completed with other additional methodologies so as to optimize the project management practices used in the core proposed approach.

Presenting an extended project management framework that combines I-Tropos with Scrum allows scaling the software system to-be around the long-term values of the company developing the software product.

Information systems' iterative development has been widely used for the last years, notably through the RUP and agile methods. Iterative development is a building process used to deliver the functionality of a system in a series of releases of increasing completeness (i.e. I-Tropos). Each release is developed in a specific, fixed time period called an iteration (i.e. Scrum). It has become a best practice for building large-scale user-intensive systems due to its potential benefits mainly *resolving major risks* before making large investments, *enabling early user feedback*, *testing* and *integrating modules and components* in a *continuous* way, and focusing on *project short-term objective milestones*.

In detail, one of the objectives of combining I-Tropos and Scrum was to provide a hybrid approach that maximizes I-Tropos and Scrum strengths to deliver a high-quality product that meets customer requirements and expectations. I-Tropos strengths provide structure and guidance through the entire software development process; allow easily adapting the process to the project needs. Although I-Tropos involves project management activities, it does not indicate how to organize the daily activities during the software development process. In contrast, Scrum strengths include the management of the day-to-day activities to organize the software development process and respond to changing requirements and feedback. This hybrid solution provides a collaborative environment during the Setting, Blueprinting, Building and Setuping phases.

In order to continue the research for this work a few suggestions for further work can be cited. For example, an exhaustive framework for managing Agile Software Product Lines (ASPL) will be developed.

9. References

1. Wautelet Y., Kolp M., Business and model-driven development of BDI multi-agent systems. Neurocomputing 182: 304-321 (2016).

2. IBM, The Rational Unified Process, Version 7.0.1 (2007).

3. Shuja A., Krebs J., IBM^{R}, rational unified process^{R}; reference and certification guide: solution designer, 1st Ed., IBM Press, 2007.

4. Gibbs R.D., Project Management with the IBM^{R} Rational Unified Process^{R}: Lessons From The Trenches, IBM Press 2006.

5. Tropos Project, "The Tropos Methodology". http://www.troposproject.org/. (accessed on October 2016).

6. Wautelet Y., A goal-driven project management framework for multi-agent software development: The case of I-Tropos, Ph.D. thesis, Université atholique de Louvain (2008).

7. Kenneth S. R., Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison-Wesley Professional, Upper Saddle River, NJ (2012).

8. Cohn M., Succeeding with Agile: Software Development Using Scrum, Addison-Wesley Professional, 1st ed. (2009).

9. Introduction to OpenUP http://epf.eclipse.org/wikis/openup/ (accessed on November 2016).

10. Shore J., Warden S., The Art of Agile Development: Pragmatic Guide to Agile Software Development, O'REILLY, 1st Edition, (2007).

11. Goldberg A., and Kenneth S. R. Succeeding with Objects: Decision Frameworks for Project Management. Addison-Wesley Professional. (1995).

12. Castro J., Kolp M., Mylopoulos J. Towards requirements-driven information systems engineering: the Tropos project (2002).

13. Wautelet Y., A goal-driven project management framework for multiagent software development: The case of i-tropos, Ph.D. thesis, Université catholique de Louvain (2008).

14. Wautelet Y., Kolp M., Achbany Y. S-Tropos: An Iterative SPEM-Centric Software Project Management Process? Working Paper, Université catholoique de Louvain, ILSM, (2006).

15. Wautelet Y., Kolp M., and Poelmans S. Requirements-Driven Iterative Project Planning. ICSOFT (Selected Papers): 121-135 (2011).

16. Wautelet Y., and Kolp M., Goal Driven Iterative Software Project Management. ICSOFT (2) 2011: 44-53, (2011)

17. J. Castro, M. Kolp, L. Liu, A. Perini, Dealing with complexity using conceptual models based on tropos, in: A. Borgida, V. K. Chaudhri, P. Giorgini, E. S. K. Yu (Eds.), Conceptual Modeling: Foundations and Applications, Vol. 5600 of Lecture Notes in Computer Science, Springer, pp. 335-362 (2009).

18. Wautelet Y., Representing, modeling and engineering a collaborative supply chain management platform, IJISSCM 5 (3) 1-19, (2012).