POLITECNICO DI TORINO

Corso di Laurea Magistrale

in Engineering and Management

Tesi di Laurea Magistrale

The Hybrid Application of Agile Project Management in IT Banking System: Optimization of Traditional Framework

Relatore

prof. Alberto De Marco

Candidato

Vittorio Amedeo Zanalda

Anno Accademico 2018/2019

Alla mia famiglia,

Table of Contents

ABSTRACT

This thesis' project is based on the awareness of the potentiality of the Modern Project Management with respect to the Traditional Project Management, which is the most diffused approach for the governance of the software development in IT Banking System. The aim of the thesis is therefore to optimize the traditional Waterfall Project Management framework by means of the hybrid application of Agile Project Management in the software development life cycle related to the bank industry.

The present work provides a theoretical part that introduces the reader in the Project Management concepts. It first focuses on the Waterfall methodology by describing the phases' flow, the benefits and the drawbacks related with this traditional approach. Secondly, the thesis aims attention at the modern techniques of Project Management known as Agile methodologies where the focal point is the Scrum Agile. Then, the differences in terms of advantages and disadvantages between the Waterfall Project Management and the Scrum Agile are critically shown in a study concerning quantitative Key Performance Indicators and qualitative Delta.

Hence, the thesis switches from theory to practice by taking into consideration the case study of the banking Information Technology framework. The banking IT structure is described as well as the architecture of the IT unit subject of analysis, named IT-Alfa. In order to seek the thesis' objective, the current study deep dives into the Waterfall Software Development Lifecycle of IT-Alfa by providing businesslike examples of activities carried out by IT-Alfa. This analysis focuses on the dynamics of project governance in terms of planning, monitoring and problem management for each of the activity's typologies described in the project portfolio section.

In comparison of the banking IT Traditional Project Management framework, the thesis' project structures the application of Agile Project Management in such environment aiming at evaluating the KPIs' trend in both the cases and at quantifying the relative benefits and drawbacks. Then, the results of such pure application of the Scrum Agile with respect to the Waterfall approach are critically analyzed. Finally, the thesis introduces the Hybrid Agile aimed at providing an effective hybrid application of the Agile Project Management combined with the Waterfall Life Cycle principles that optimizes the traditional Project Management framework of IT-Alfa.

To conclude, the final chapter is reserved for the achievement and discussion resulting from the development of the thesis, concluding with a paragraph about possible future developments on the topic.

1. Introduction

In this chapter, the aim of the thesis will be described. The problem will first be introduced by giving an overview of the framework subject of analysis in order to explain the objective this study seeks to achieve. Then, the purpose of the thesis is reported through the presentation of the case study that will be analyzed in detail in the dedicated section. Lastly, the structure of the document is reported to better understand the conceptual plan.

1.1 Problem Statement

Project Management is an essential activity for large, medium and even small-scale enterprises that need for cost, time and human resources planning and, as a matter of fact, the application of an effective Project Management methodology results in higher risk mitigation with elevate performance activity. Project Management is not new; it has been in use for hundreds of years for constructions as the Great Wall of China or the Pyramids of Giza. Traditional Waterfall project methodologies have been the most diffused approach to Project Management aiming at the implementation of complex resource organization projects. However, the effectiveness of the Project Management technique strongly depends on the framework of the project and, by taking into consideration the Software Development framework, the Waterfall approach has been gradually replaced by iterative processes as the Agile Project Management.

The framework of this study is precisely the Software Development Life Cycle. In detail, the thesis concerns how the evolution of the Project Management methodologies affects the IT structure of banking industry. Financial institutions are by definition low-risk and profit business organizations. Banks need for a structure designed in order to mitigate as much as possible any type of risk and to perform their business in a productive way. Since the large-scale banks have an internal IT department, this means that not only the brand-new IT enterprises should apply the cutting-edge Project Management approach as Agile, but also the banks themselves.

Here the problem comes. On one hand Agile system development methodologies, even if are gaining popularity due to the successful functionality realized by many worldwide organizations, demonstrated a low applicability and adaptability to many environments. On the other hand, banks have many older systems and they have been among the most conservative and among the slowest to adopt new technology and new methodologies for software delivery. Moreover, financial sector is heavy regulated and Agile Project Management requires a cultural change and mindset transformation for the whole company.

Therefore, this thesis' project will face the presented critical issue by means of a state-of-art literature review and a case study application giving a solving process of a hybrid approach to Agile technique for the IT banking department.

1.2 Study Objective

The final objective of the thesis is to define a hybrid application of the Agile Project Management combined with the Waterfall Life Cycle in IT Banking System. To achieve this target, the Waterfall and the Agile methodologies are first introduced and analyzed aiming at studying the relative benefits and drawbacks. Then, a real case study related with the financial world is taken into examination, where the Bank has an internal IT Service, named IT-Alfa, that works with Traditional Project Management following the Waterfall approach in order to satisfy specific Business Requirements by developing new software and buying new hardware or by strengthen the existing software and hardware.

The challenge is to optimize the current traditional framework of the IT-Alfa by studying a synergy of the already applied Waterfall approach with the Agile Project Management. Since both the methodologies will be explained, the thesis aims at implementing the key benefits of the different techniques in real projects' examples by defining the as-is situation and the to-be management. The final purpose is to identify a hybrid application of the Agile Project Management in the Bank that guarantees not only an increase in terms of project performance but also a correct integration in the IT structure of the Bank.

1.3 Structure of the Thesis

The summary below shows the articulation of the thesis divided into the following sections.

The first part (chapter 2) reports the theoretical steps necessary to better understand the phases and the processes of both Waterfall and Agile methodologies as well as their advantages and disadvantages, their differences and their synergy. The concepts that will be described are the combination of the candidate's background, raised thanks to the Project Management course held by the thesis' supervisor and due to work experience, and a state-of-art literature review focused on papers concerning different kinds of applications and uses of Project Management methodologies based on both real and hypothetical cases.

The second part (chapter 3) is the most significant section of the thesis, as well as the most substantial. This chapter will show the hybrid application of Agile Project Management in the real case study concerning the IT department of the Bank that applies the traditional Waterfall approach. To achieve this purpose, the banking IT framework is first introduced followed by the description of the project portfolio that will be investigated. Then, practical examples of the Waterfall life cycle used by IT-Alfa will be described in order to show the pros and cons with respect to the application of Agile Project Management by taking into consideration the outcome of chapter 2. Both the situation will be analyzed through the evaluation of some quantitative KPIs in order to determine which Project Management approach is the most performant in IT-Alfa. Furthermore, the hybrid combination of the Agile technique with Waterfall methodology for IT-Alfa projects will be analyzed in order to study how to optimize the traditional framework by introducing Agile concepts with hypothetical example.

Finally, the last section (chapter 4) is reserved for the conclusion and discussion resulting from the development of the thesis, concluding with a paragraph about possible future developments on the topic.

2. Project Management Evolution: from Waterfall to Agile

This chapter represents the theoretical part of the thesis. First, the big picture of Project Management is presented as well as the framework object of analysis: Software Development Life Cycle. The topic is then introduced by defining the feasibility study after which the Project Management take places with different characteristics. On one hand, the traditional approach to Project Management is showed by analyzing the Waterfall method, its phases and its pros and cons. On the other hand, the Agile Project Management is illustrated by analyzing the Scrum methodology and all its features. Finally, the two techniques are compered by evaluating KPIs and delta.

2.1 Project Management

Project Management is defined as “the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements” by the Project Management Institute. As a new project begins, Project Management process starts. Independently from the Project Management methodology chosen, every project has a preliminary phase of business requirement definition followed by the project evaluation and its cost approval. This initial step will be accurately described in the following paragraphs, but first let’s introduce the framework this thesis focuses on.

2.1.1 Software Development Life Cycle

Software Development Life Cycle (SDLC) is the process of planning, developing and deploying software. The SDLC aims to produce a high-quality product to deliver IT solutions that meet business need and exceed customer expectations, reach completion within times and cost estimates.

By giving a formal example related to the banking industry, the business department of a bank, or of another financial institution, has the job of making data computation for evaluating the economic trend of the company (e.g. risk mitigation, accounting balance, other). Therefore, to guarantee the correct elaboration of such data there exists an IT department, internal or external to the institution, that maintains the already existing software and develops new software functionalities. These new software features represent the heart of the Software Development Life Cycle as they are the input the business unit requires and the output the IT structure provides by software developing. The way this development process is performed strongly depends on the Project Management approach. The thesis will concern the Waterfall methodology in paragraph 2.2 and then the Agile technique in paragraph 2.3.

Before arguing about the different types of SDLC, the next paragraph will describe the initial study of a new project aimed at defining its technical and economic feasibility.

2.1.2 Feasibility Study

The Feasibility Study represents the complex and delicate first step of a new project where the Project Manager needs to perfectly understand what specific necessity is required by the Client for each part of the business structure as well as the effort needed to achieve the purpose of the project.

The starting point is the Business Requirement Book (BRB), the document that reports the objectives that the final User, or the Client, requires for the implementation of the project. At the very beginning the BRB gives only a high-level description of the business requirements, therefore the stakeholders perform a Feasibility Study aimed at in-depth understanding the objectives and the scope of the project as well as the cost and benefit relationship.

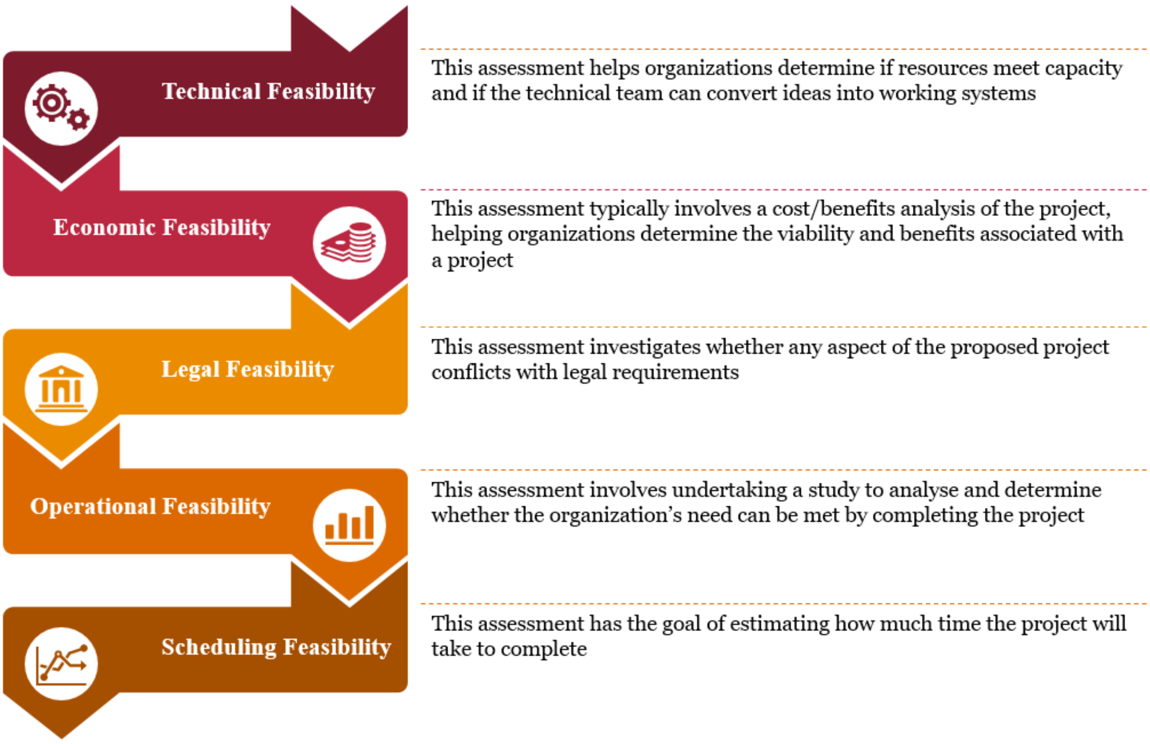

Figure 1: *The five areas of Feasibility Study*

The Feasibility Study determines whether a project is worth the investment by the investigation of five different areas [1]:

1. **Technical feasibility** is the most important area of analysis and focuses on the technical resources, both tangible and intangible, available to the organization. It helps the project managers to determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves evaluation of the hardware, software, and other technology requirements of the proposed system.
2. **Economic feasibility** concerns cost/benefit study-helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. The final cost is usually based on the estimated effort of the internal and external resources plus the potential expense of hardware and/or licenses.
3. **Legal feasibility** investigates whether any aspect of the proposed project conflicts with legal requirements like local laws, data protection regulations, or social media acts.
4. **Operation feasibility** determines whether and how well the organization's needs can be met by completing the project. Operational feasibility studies also analyze how a project plan satisfies the requirements demanded by the clients.
5. **Scheduling feasibility** determines the time horizon the project will take place by assuming a time-bound specifying when the results can be achieved.

It is important to specify that the effort spent for the feasibility study strongly depends on the relevance of the project. It means that, on one hand, some projects are very complex and relevant so that the Business Owner can pay for a thorough Feasibility Study. On the other hand, minor projects do not have budget for this analysis and, as a consequence, a macro-estimation of impacts and costs is performed by the Project Manager. In the latter case, any detailed study about the requirements is performed in parallel with the development of the project.

One can easily imagine that the aforementioned five areas of feasibility are strictly correlated because, for instance, when the system requires meaningful technical processes, the cost will increase as well as the time horizon will be widened. Then, there is the economic feasibility, a key parameter that determines whether the project is worth the investment. If the estimated cost overcomes the budget, or the project cannot be performed, or a cheaper technical/ operational solution is adopted by eventually restricting the requirements' scope.

Sometimes it may happen that even if there is the budget required, the analysis of the technical and/or operational feasibility has negative result. In this case, the solution may not be simple and smooth, and the Project Manager must carry on an in-depth study by usually taking on new external resources, either tangible or intangible.

Concerning the legal and the scheduling feasibility, they should rarely stop the project because the legal constrains determine the boundaries of the project, whereas the planning can meet the Client desiderata in time or in advance with a potential bonus, otherwise in case of delay a penalty is usually applied.

Therefore, after these five criteria had been analyzed, evaluated and then approved, the project can take place.

2.1.3 Scheduling the Activities

As the preliminary phase described in the paragraph above is successfully completed, the project starts by defining the operational plan that every working team hired must follow and meet.

The purpose of the planning procedure is to establish a well-defined structure of the activities' flow by setting the milestones and the deadlines aimed at organizing in detail the work across the teams.

The way this task is performed radically changes according to the methodology the Project Manager applies. The following paragraphs will first describe Waterfall methodology (paragraph 2.2), that symbolizes the Traditional Project Management, and then the Agile methodology (paragraph 2.3), that represents the Modern Project Management. The scheduling process, therefore, will be discussed by two very different points of view.

2.1.4 Monitoring the Activities

As the development of the project begins, the Project Manager must control the trend of the activities' flow so he or she monitors the performance of the working force.

An effective monitoring of the activities may guarantee on time delivery and cost-effectiveness and it can be achieved not only by controlling that every teams meet the preset deadlines, but also by performing an accurate and dynamic risk analysis that evolves over time.

As well as the scheduling procedure, the monitoring process varies according to the project management technique chosen by the Project Manager. For this reason, paragraph 2.2 will show the monitoring applied in Traditional Project Management by means of the Waterfall approach, whereas paragraph 2.3 will describe how control practices are performed in the Agile Project Management.

2.2 The Waterfall Methodology

The concept of the Waterfall methodology was born in 1970 by Winston W. Royce who elaborated this approach in order to face the development of complex software systems and, nowadays, the Waterfall process represents a project management guideline widely recognized and applied in companies' managerial practice across many industries besides IT.

2.2.1 The Waterfall Life Cycle

The Waterfall approach is based on a linear sequence of phases where each phase is completed and approved before the team moves on the next one and backward moves are not allowed [2].

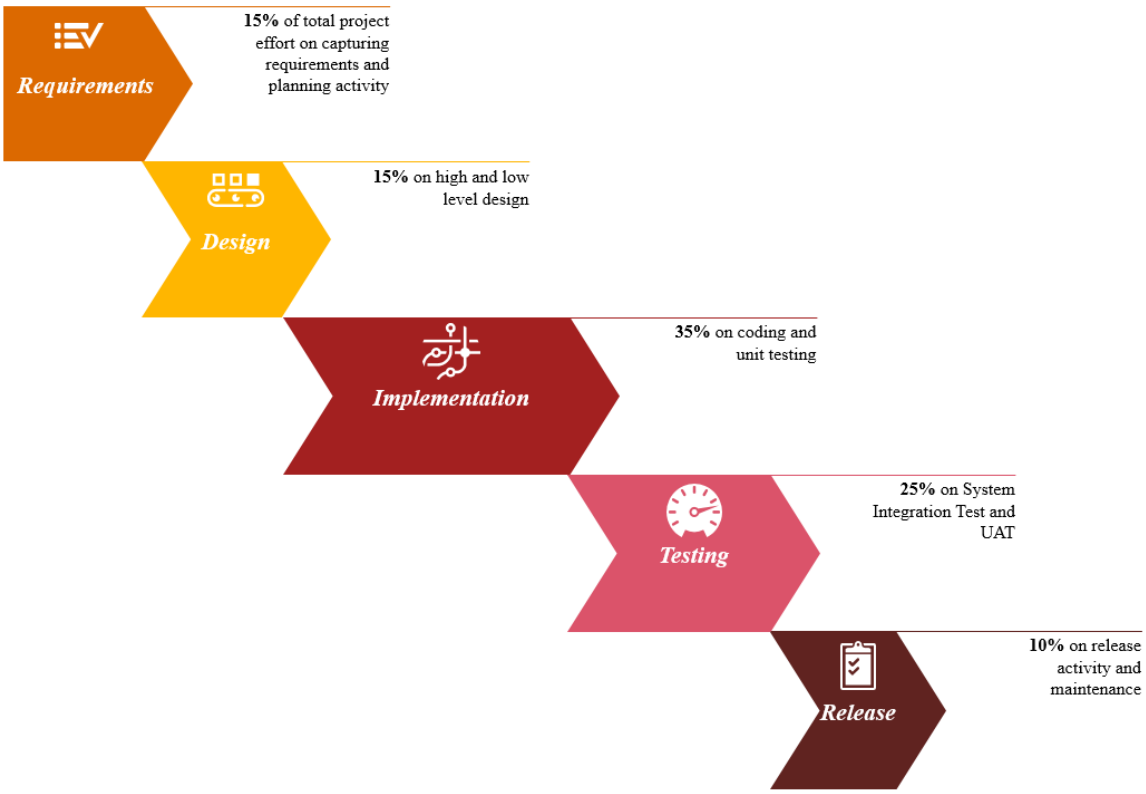

Figure 2: *The phases of Waterfall Life Cycle*

The figure above (Figure 2) shows the five fundamental phases of the Waterfall lifecycle in the progression they must be carried on by the working teams.

1. The **Requirements** phase is usually completed by carrying out a feasibility study as described in the paragraph 2.1.2. This step concerns the definition of the objective, the scope and the implementation boundary of the project as well as the estimation of the effort of time and cost needed to meet these requirements. In the Waterfall framework is very important to deeply understand all the aspects the project requires because applying a restatement and a change after the beginning of the implementation phase may result in higher cost and extension of time;
2. The **Design** process is strongly related with the requirements one. The system's designer makes use of the information collected in the Requirements phase, usually stated in a document shared and consolidated by both the User (or Client) and the working team, in order to design the software and, potentially, the hardware systems [3]. This phase must guarantee the fulfillment of the requirements by transforming them on the specifications of definite hardware and software technologies. Therefore, the output of the design process is to be the guideline the developers uses to implement and develop the project;
3. **Implementation** begins when the first two phases described above are definitely completed. The implementation process is often the longest part and it concerns the transformation of the design specification into code writing. Aiming at lowering operational risks, the analysts should perform heuristic evaluation and test of usability and applicability of the new product meanwhile they are programming. Doing so, the implementation phase results to be well consolidated and the project can move to the next step;
4. **Testing** aims at ensuring that the project is meeting customer expectations. The User Acceptance Test (UAT), performed by the Client over the output of the implementation phase, aims at identifying and correcting any possible error. Only when the UAT is successfully completed, the project is ready for the final phase;
5. **Release** process usually comes with maintenance phase. The project is rolled out to the Client who starts using the developed application. As issues are found due to improper requirements determination or other mistakes in the design process, or due to scare data quality, software changes are planned and released to rectify any problem. There may be two different cases; the issue reported can be easily solved by means of a bug fixing, that means no extra cost and is not time consuming, or the User can face a complex error. In the latter case the solution may be released only after an accurate analysis that requires new effort for the completion of the project.

2.2.2 The Benefits of the Waterfall Process Development

The Waterfall methodology is a linear approach to process development that severely follows the steps described in the paragraph 2.2.1 providing the following strengths [4]:

- The developers and the User argue about the requirements and the whole project as a very first step and, as a consequence, the delivery and the project itself result clever and easier to meet. Moreover, the high attention spent for the Feasibility Study guarantees the correct allocation of effort by minimizing the wastefulness of time and cost and ensures the fulfillment of the requirements. Because the final objectives are designed in detail, all the stakeholders acquire a sound knowledge about the main desiderata and the result to achieve;
- The Waterfall method allows saving money and time. The developers focus only on what is necessary and they may correct possible error in the design phase or implementation phase before the UAT is carried out. Thanks to the definition of the objectives and the scope of the project during the feasibility study, both the analysts and the clients know the boundaries in financial terms and as time horizon;
- The Waterfall approach perfectly works with teams who operates nearby because it offers well-defined assignments and tasks that allow working teams to schedule time and effort in an effective way. The organization of the team members is therefore straightforward to plan and monitor;
- The Waterfall methodology, according to which each phase is completed and approved before the team moves on the next one, secures a high-level control on the production process and on the related risks because it preserves the big picture of the whole work and its features. It is effective when governing and managing potential findings and critical issues on complex project.

2.2.3 The Drawbacks of Waterfall Life Cycle

On one hand, it has been shown as the Waterfall technique of moving step by step toward ultimate release leads to advantages in terms of Project Management, however, on the other hand, the Waterfall constrain according to which backward moves are not allowed may represent issues for the completion of the project. There follow the main disadvantages [5]:

- Not going back to previous phases puts huge pressure to the planning. As problems come, not only the delay is a critical issue, but also the new project scheduling that may ensure to solve the errors identified and not to waste the effort already spent;
- When delay is collected in one single step, all the next phases are affected resulting in a delivery postponement;
- When UAT has negative results, it may be critical solving the issues identified by the customers because the testing process is at the end of the Waterfall lifecycle and just before the release. Therefore, or a new implementation phase begins by postponing the delivery, or the project closes in time but not fully meeting the requirements and a new evolutionary process is then scheduled. This situation means extra time and cost;
- Since the requirements are defined at the beginning, the developers are facilitated to design and to implement the product as agreed; however, this does not necessarily mean that the working team satisfy customers' desiderata because, sometimes, the client gets a sound knowledge of the product features only by performing the UAT and, as a consequence, new specifications are required. This is the case of a Change Request (CR), which concerns a project follow-up with new budget and a new delivery deadline.

2.3 The Agile Methodology

Agile is a framework of project management and development based on an iterative and incremental process encouraging rapid and flexible responses to change. The requirements, and the relative solutions, born and evolve during the whole duration of the Agile process thanks to the collaboration of self-organized, multidisciplinary and cross-disciplinary teams. Moreover, Agile methodology promotes adaptive planning and on-going developing procedures where the software release must work in a continuous, rapid and flexible way such that the teams are able to meet the changes due to the market or required by the User.

2.3.1 The Agile Manifesto

Agile is effectively an approach rather a single methodology, because there exist many Agile methodologies that have uncovered a better way to deliver value and all Agile methodologies share the common vision and the core values presented in the Agile Manifesto.

The Manifesto for Agile Software Development was the result of the meeting taken in 2001 by seventeen process methodologists who formally wrote the four values and the twelve principles that found the basis of Agile framework.

The Manifesto reads [6]:

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

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

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

(Beck, K., et al. (2001) The Agile Manifesto)

Then, the Manifesto reports the following principles [6]:

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

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

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

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

Build projects around motivated individuals.

Give them the environment and support they need,
and trust them to get the job done.

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

Working software is the primary measure of progress.

Agile processes promote sustainable development.

The sponsors, developers, and users should be able
to maintain a constant pace indefinitely.

Continuous attention to technical excellence
and good design enhances agility.

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

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

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

(K. Beck, et al. (2001) The Agile Manifesto)

The following image (Figure 3) gives a graphic and concise view of Agile's values.

Figure 3: *The Agile Project Management values*

2.3.2 The Agile Development Process

Agile approach delivers potentially shippable product in short iterations following business priorities, while providing the ability to deal with uncertainty and adapt to changing requirements. The phases are almost the same identified in the Waterfall methodology, but they are cyclic rather than sequential as shown in the image below (Figure 4).

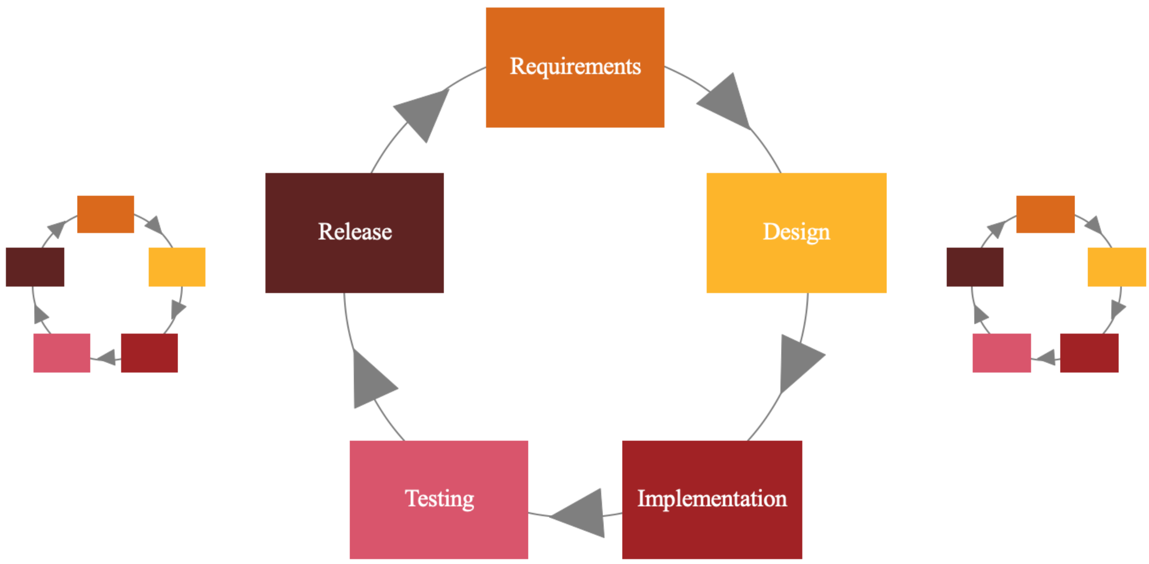

Figure 4: *The phases of Agile Life Cycles*

Each cycle, which duration may range between 1 and 4 weeks, is called sprint and each sprint has specific goals, planning, execution and release. The idea is to delivery multiple times to the Client by getting continuous feedback aiming at improving and completing the final product. As a result, this on-going coming out of new requirements definitely decreases the failure risk of the project by meeting the necessity and the specification after each iteration and by releasing incrementally value added.

The way the sprints are organized and carried out depends on the framework used; the Agile methodology is indeed difficult to standardize because of its adaptive nature. Scrum, one of the most used Agile frameworks, emphasizes management and business involvement on a day-day basis; having all key stakeholders together on a regular basis provides high quality of work, increased visibility, and adaptability. However, other Agile frameworks exist such as Lean, Kanban, XP (Extreme Programming), AUP (Agile Unified Process), ASD (Adaptive Software Development), DSD (Dynamic System Development), SAF (Scalable Agile Framework), DAD (Disciplined Agile Delivery), FDD (Future Driven Development), or even hybrids (Scrum + XP + Kanban), and essential practices such as test-driven development, test automation, continuous integration and continuous delivery which help increase quality and visibility into the deployment pipeline.

The following Agile methodologies are the most commonly applied:

- **Scrum** aims at applicability for managing and controlling iterative and incremental projects of all types;
- **Lean** is used when the project focuses primarily on the elimination of waste;
- **Kanban** secures emphasis on just-in-time delivery and limiting the Work-In-Progress (WIP);
- **XP** is an engineering disciplined approach for delivering high-quality software quickly and continuously.

The pie chart below (Figure 5) shows the big picture of the trend of Agile methodology.

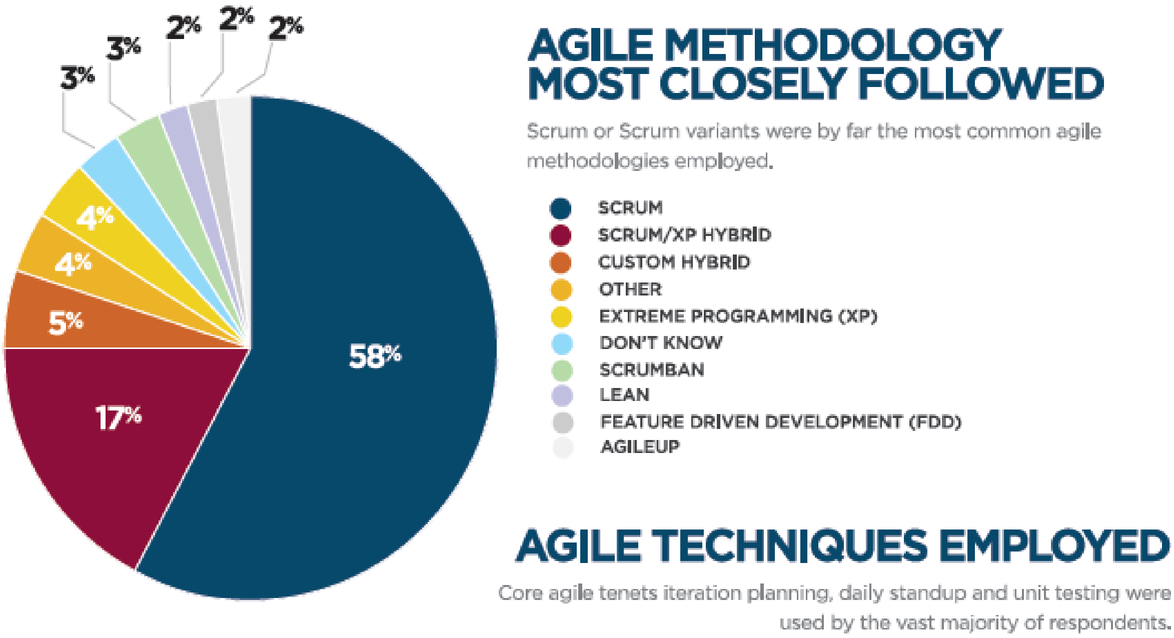

Figure 5*: Agile methodologies most diffuse [7]*

Even if combining different frameworks or aspects of different frameworks can provide significant advantages and help accelerate the maturity of Agile in an organization, the thesis focuses on the Scrum methodology. Because Agile Scrum not only has the widest diffusion, as reported in Figure 5, but also it represents strong similarity and compatibility with the Waterfall methodology and, as a consequence, Agile Scrum is the best approach to be combined with the traditional framework.

2.3.3 The Scrum Agile

The Scrum can be described by taking into consideration the quote «Agile is a mind-set defined by values, guided by principles and manifested through many different practices» by Ahmed Sidky. Therefore, this paragraph will first introduce the mind-set of the Scrum and secondly shows the iterative approach.

The Scrum theory is based on three pillars:

1. **Transparency:** significant aspects of the process must be visible to those responsible for the outcome;
2. **Inspection:** Scrum users must frequently inspect Scrum artefacts and progress toward a goal to detect;
3. **Adaptation:** to make adjustments as soon as possible to minimize further deviation.

The following five values represent the Scrum philosophy:

1. **Courage** to make the right thing and work on tough problems;
2. **Focus** on the work of the Sprint and the goals of the Scrum team;
3. **Commitment** to achieve the goals of the Scrum team;
4. **Respect** each other to be capable and independent people;
5. **Openness** to be open about all the work and the challenges with performing the work.

2.3.4 The Scrum Roles

The Scrum identifies only three key typologies of actors that are needed for the framework to work correctly.

- The **Product Owner** decides which features and functionality to build and the order in which to build them. He or she is the responsible for defining the characteristics that are needed in the project;
- The **Scrum Master** acts as coach and facilitator to Scrum teams, ensuring that the team and the rest of the organization obtain optimum results from the Scrum process. He or she is responsible for protecting the team and the process running the meetings and keeping work going;
- The **Development Team** is a diverse, cross-functional collection of all of the types of people needed to design, build, and test a desired product. Team members often play multiple roles because someday developers may test, and other days testers may develop; this means the team needs not only to be cross-function but also self-organized to work with synergy. The team is responsible for transforming the features of the product on potentially shippable deliverables. The optimal team dimension ranges from 5 to 9 analysts. On one hand, if the team members are lower than 5, the iteration decreases as well as the level of productivity and, moreover, there may face lack of skills. On the other hand, a team with more than 9 specialists results complex to manage, especially in Scrum framework.

The operative roles will be evident after describing the work cycle of the Scrum, as well as the term ‘potentially shippable’ will be clearer.

2.3.5 Scrum Artefacts

The Scrum framework concerns 3 artefacts, represented in the figure below (Figure 6) and then described.

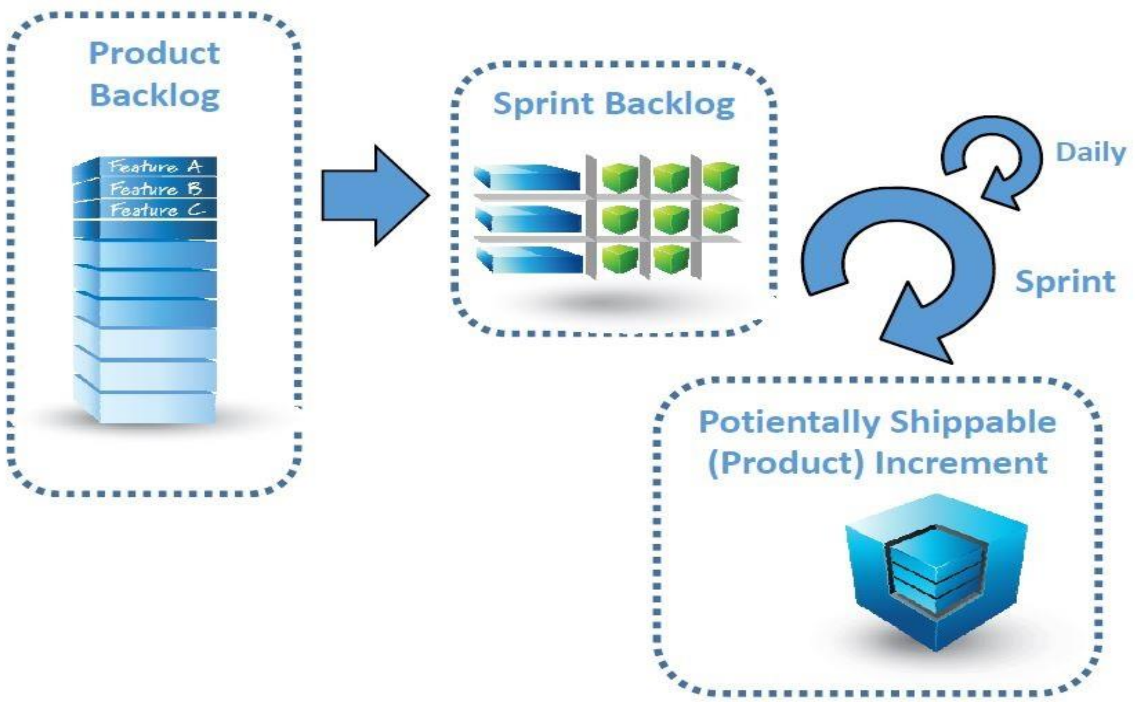

Figure 6: *The three artifacts of Scrum Agile*

The **Product Backlog** is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product. It is an alive entity and changes frequently according to the Product Owner decision. The Product Owner is indeed the responsible of the Backlog but also the Scrum Master and the Development Team may help to build it.

The following list concerns the different types of items that characterize the Backlog:

- User Stories (described in business language);
- Technical Stories;
- Bugs;
- Rework/ Refactoring;
- Documentation Needs;

Product Backlog items have the attributes of a description, order, estimate and value as well as a different size, depending on the moment in which they will be developed.

The act of adding detail, estimates, and order to items in the Product Backlog is called Refinement and it is an ongoing process in which the Product Owner and the Development Team collaborate on the details of Product Backlog items. Items are reviewed, revised, decomposed and estimated by actually no more than 10% of the capacity of the Development Team and the Scrum Team decides how and when refinement is done.

There are solid rules to follow for ordering the Product Backlog features:

- Higher ordered Product Backlog items are usually clearer and more detailed than lower ordered ones;
- More precise estimates are made based on the greater clarity and increased detail; the lower the order, the less detail;
- Product Backlog items that will occupy the Development Team for the upcoming Sprint are refined so that any one item can reasonably be "Done" within the Sprint time-box;
- Product Backlog items that can be "Done" by the Development Team within one Sprint are deemed "Ready" for selection in a Sprint Planning;
- Product Backlog items usually acquire this degree of transparency through the above described refining activities.

When the Product Backlog has been built and prioritized, the **Sprint Backlog** can begin. The Sprint Backlog is the set of Product Backlog Items (PBIs) selected for the Sprint, plus a plan for delivering the product Increment and realizing the Sprint Goal.

Sprint Backlog may be organized and managed with a Kanban board, often referred as Sprint Board or Scrum Board, because it makes visible the work to be done by the Development Team and it helps the team to organize the work by setting the PBIs ordered by priority from top to bottom depending on the story points estimated in the Backlog. Each feature is then divided and detailed in a set of tasks that can measure the effort needed to meet these requirements. The figure below (Figure 7) provides a graphic view of this process.

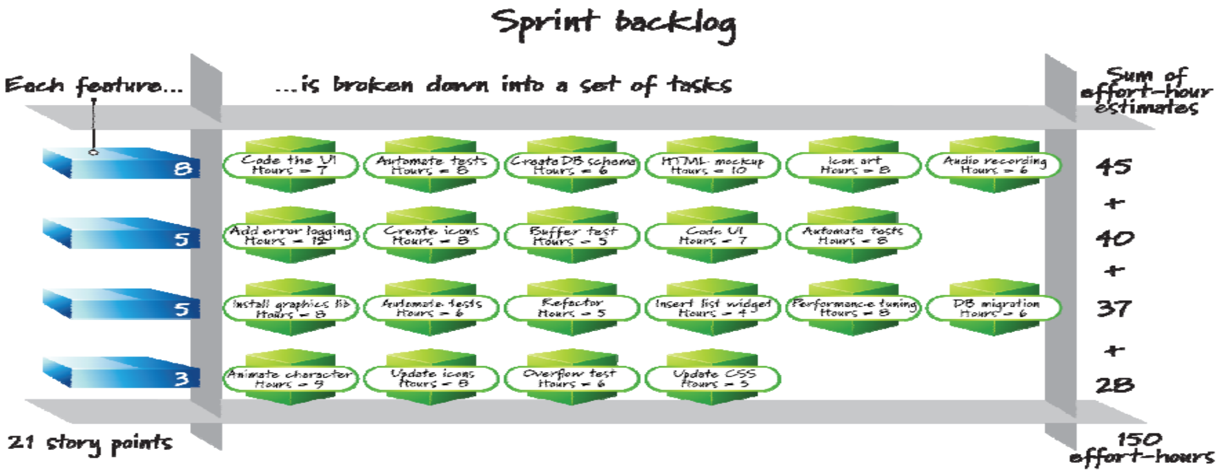

Figure 7: *The Sprint Board showing the process of the Sprint Backlog*

In order to monitor the evolution of the sprint backlog, the Burn Down chart is performed. The Burn Down chart displays the amount of remaining work to reach a goal and shows the trend line that should be followed to finish all the work. It is used by both the Product Owner to track the total work remaining and by the Development team to track the total work remaining at least for every Daily Scrum to project the likelihood of achieving the Sprint Goal. Moreover, this tool is applied in Sprint Review to assess progress toward completing projected work by the desired time for achieved the objectives.

Finally, there is the potentially shippable product **Increment** as the sum of all the Product Backlog items completed during a Sprint and the value of the increments of all previous Sprints. At the end of a Sprint, the new Increment must be “Done”, which means it must be in useable condition and meet the Scrum Team’s definition of “Done”. One says ‘potentially shippable’ because the Product Owner must confirm the release; anyway, the Increment must be in useable condition regardless of whether the Product Owner decides to actually release it.

When the Product Backlog item or an Increment is described as “Done”, everyone must understand what “Done” means. Although this varies significantly per Scrum Team, members must have a shared understanding of what it means for work to be complete, to ensure transparency. The “Definition of Done” for the Scrum Team is therefore used to assess when work is complete on the product Increment and the same definition guides the Development Team in knowing how many Product Backlog items it can select during a Sprint Planning Meeting. The figure below emphasizes the process described.

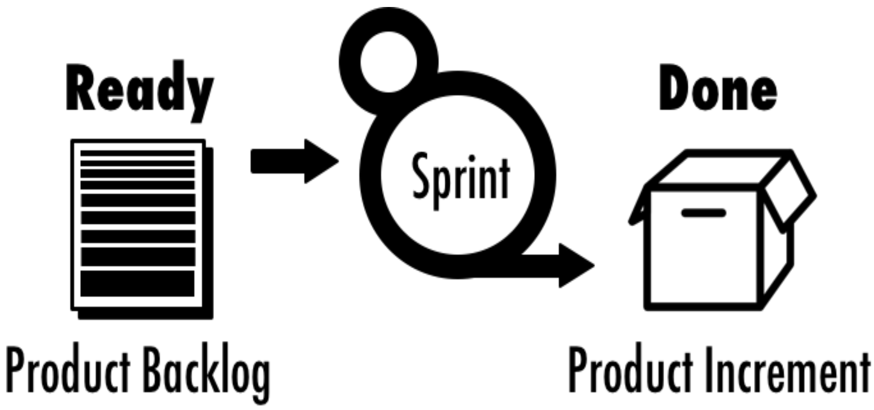

Figure 8: The Product Backlog Items process in Scrum framework

2.3.6 Scrum Events

The Agile Scrum project is achieved by planning, implementing and releasing many sprints. The **sprint** is therefore the heart of the Scrum and it can be defined as a time-box of one calendar month or less during which a "Done", useable, and potentially releasable Product Increment is developed. The figure below shows a big-picture of this approach.

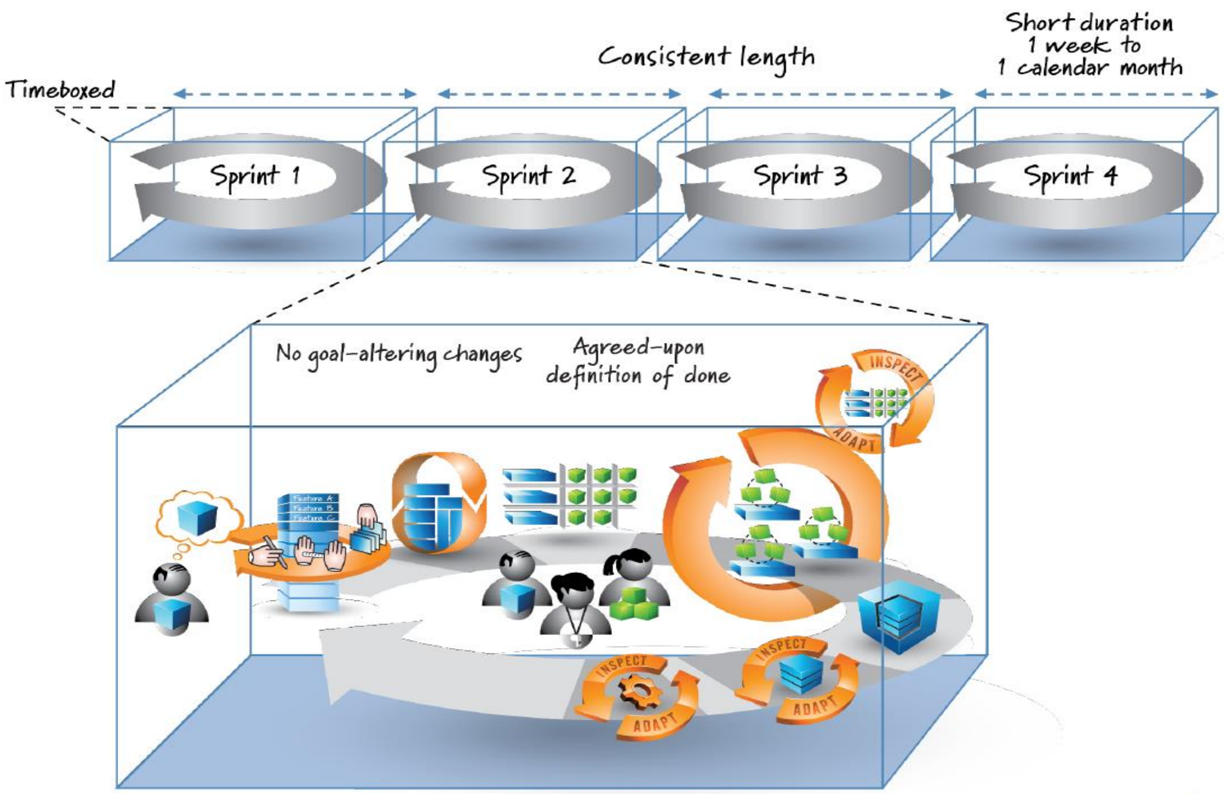

Figure 9: The big picture of the Scrum process [8]

The first step is the **Sprint Planning** where the Team plans the work to be performed. The items committed to complete are therefore selected from the Product Backlog during the Sprint Planning Meeting that usually last 8 hours for a one-month Sprint or 4 hours for a two-weeks Sprint, respectively. The initial half of the meeting concerns what will be delivered in the Increment resulting from the upcoming Sprint; the Development Team works to forecast the part of Product Backlog that will be developed during the Sprint and the Product Owner presents ordered Product Backlog Items to the Development Team and the entire Scrum Team collaborates on understanding the work of the Sprint. The second half of the meeting aims at defining how will the work needed to deliver the Increment be achieved; the work planned for the Sprint by the Development Team is decomposed in tasks, of one day or less, by the end of this meeting and the Product Owner may be present to clarify the selected Product Backlog Items and to help make trade-offs.

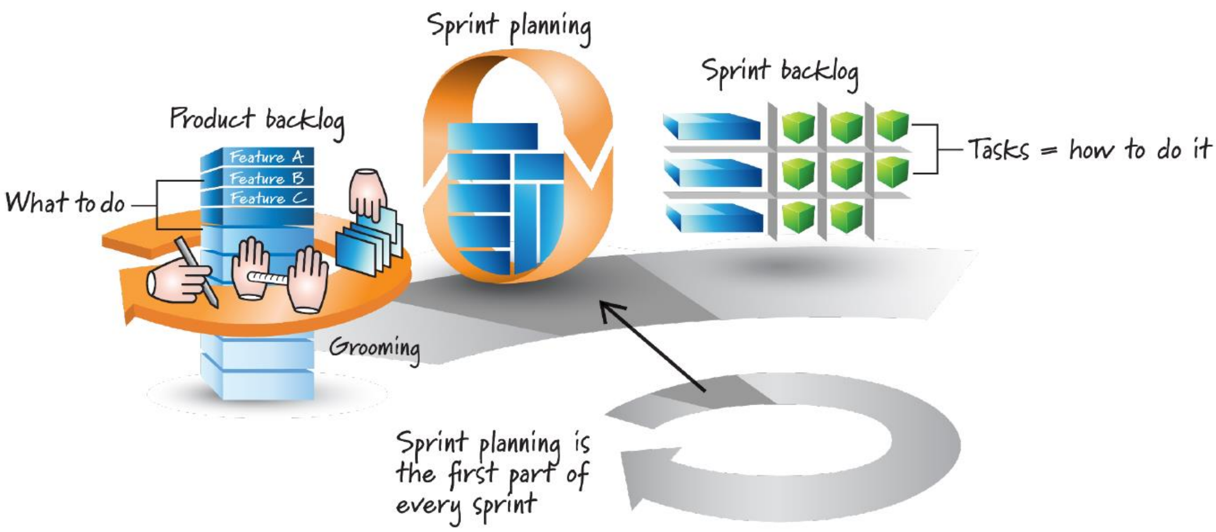

Figure 10: The Sprint Planning phase [8]

As the Sprint has been planned, **Daily Scrum** sets the context for each day’s work and helps the team stays on track. Daily Scrum is not a status meeting for the Scrum Master, but rather commitments in front of peers, a key inspect and adapt event for coordination of the teams work and team accountability towards the Sprint Backlog. This 15 minutes time-boxed meeting focuses on three points; first question is ‘What did you do yesterday?’ in order to discuss with the team the work completed the previous day, second ‘What will you do today?’ is answered to understand with the team the work planned to complete within today and, finally, current and future issues, risks, obstacles, or dependencies are investigated.

Daily Scrum is based on solid principles as it is held at the same time and in the same place every day, the 15 minutes limit must be respected, anyone can attend but only the Development Team participates, and it does not concern technical discussions nor attempts to solve issues. The Scrum Master has several duties as well. He or she must ensure that the Development Team has the meeting even if, as already mentioned, only the Development Team is responsible for conducting the Daily Scrum. Moreover, the Scrum Muster must enforce each aforementioned rule of Daily Scrum by teaching and supervising the Development Team.

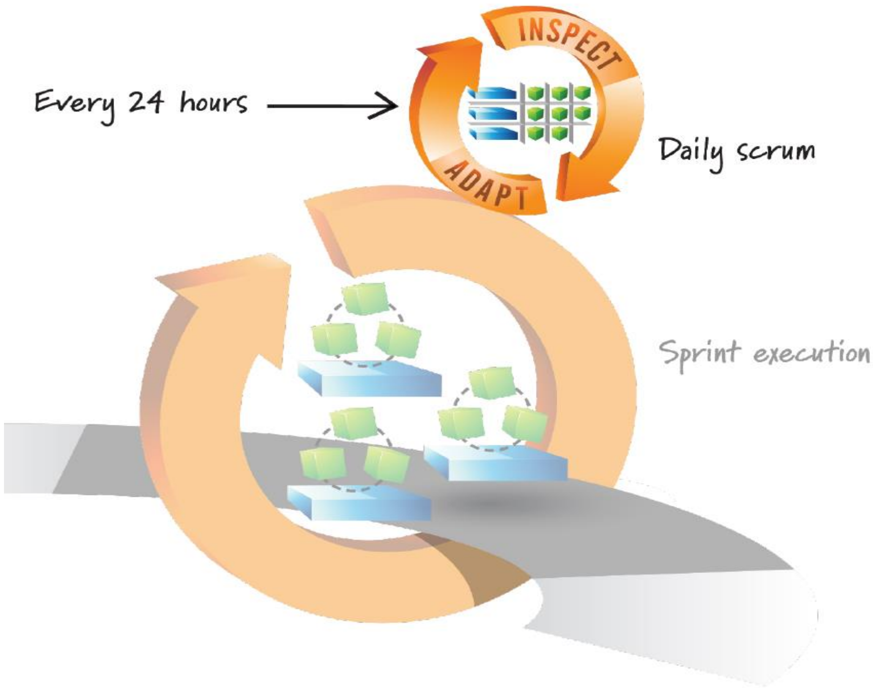

***Figure 11**: The Daily Scrum event [8]*

As next-to-last activity, the **Sprint Review** is performed by the Team demonstrating the completed work at the end of each sprint. It is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog and it is a 4 hours' time-boxed meeting for one-month Sprints. First, the Scrum Team and Stakeholders share what was done in the Sprint, so the Product Owner identifies what has been "Done" and what has not been "Done" and the Development Team discusses what worked well during the Sprint, what problems it ran into, and how those problems were solved. Second, the Development Team demonstrates the work that it has "Done" by using the Product Increment itself and not with any intermediate work product (documents, presentations etc.) and the feedback given by Stakeholders provides very valuable inputs to the whole team. Finally, the Product Owner discusses the Product Backlog as it stands as well as future work is discussed. The Review meeting also ensures a smooth introduction of the following Sprint Planning because, on the basis of what has been performed and the issues met, it is possible to forecast a more accurate completion date for the product being developed.

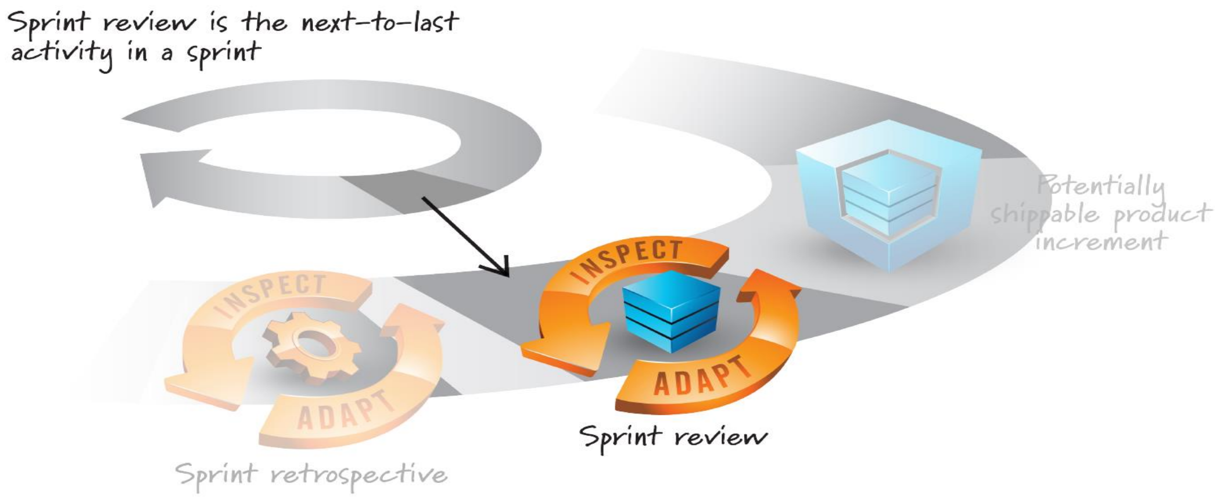

Figure 12: *The Sprint Review phase [8]*

The final step concerns the **Sprint Retrospective** that ensures Team reflects on how to improve their process and deliveries. This 3 hours' time-boxed meeting, with respect to a one-month Sprint, is an opportunity for the Scrum Team to inspect itself and adapt, in order to create a plan for improvements to be addressed as soon as possible. The aim of the Retrospective is to discuss the things the Team should continue to do, in order to identify the challenges that could improve the work and to prioritize the improvements needed for the next Sprint. Therefore, although upgrades may be implemented at any time, the Sprint Retrospective provides a formal opportunity to focus on inspection and adaptation. The Scrum Master is the key facilitator, who participates as a peer team member from the accountability over the Scrum process and teaches all to keep the meeting within the time-box.

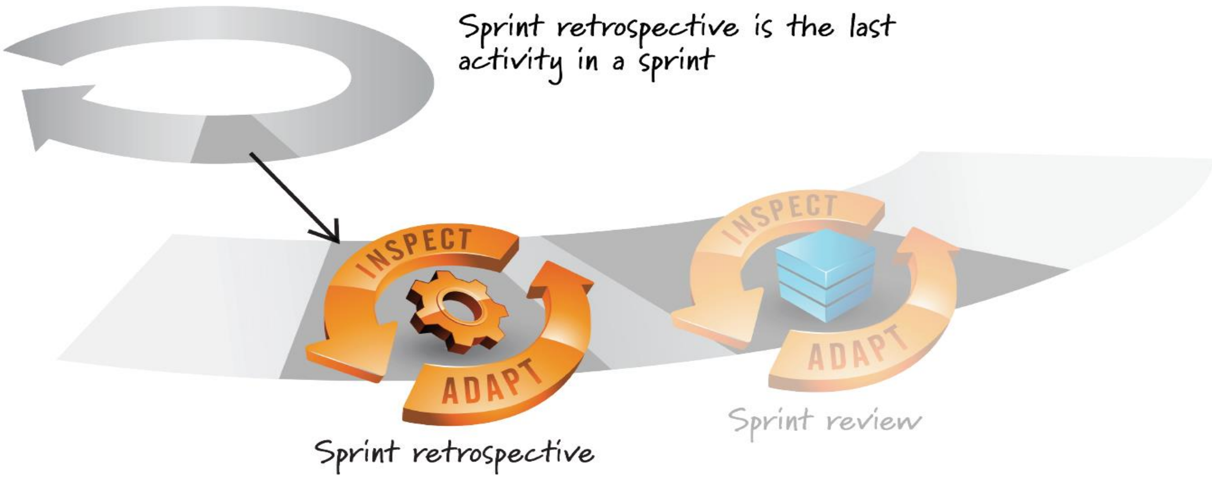

Figure 13: *The Sprint Retrospective phase [8]*

As the Sprint Retrospective is completed, a new Sprint can begin and, therefore, a new cycle starts by holding the Spring Planning meeting.

Sprint Cancellation, that is often traumatic to the Scrum Team, may represent a critic issue. Indeed, they consume resources, since everyone has to regroup in another Sprint Planning to start another Sprint. The Product Owner, with explicit agreement with stakeholders, the Development Team, and the Scrum Master, can cancel a Sprint before its completion. It should only happen under extreme conditions: whenever the Sprint Goal has become obsolete or not attainable and when the company changes direction or market, or technology conditions change. As a consequence, if part of the work is potentially releasable, the Product Owner typically accepts it and all incomplete Product Backlog Items are re-estimated and put back on the Product Backlog.

2.4 The advantages of Agile over the Waterfall Project Management

Since both traditional and modern approaches to Project Management have been introduced and explained, this section will compare the different techniques by first investigating a state of art literature review and then collecting the qualitative delta between the Agile and the Waterfall methodologies. Finally, the Key Performance Indicators are described aiming at providing a quantitative aspect to the thesis' analysis.

2.4.1 State of Art Literature Review

According to a state of art literature review, a field of studies shows the differences between the Agile and the Traditional Project Management and states their respective benefits and drawbacks. On one hand, the advantages of Agile over Waterfall are strongly affirmed in relation with software development projects; for example, D. Fernandez and J. Fernandez concluded their paper 'Agile Project Management Agilism versus Traditional Approaches' [10] writing "Agile practices, including project management, grew out of a need to manage projects characterized by complexity and uncertainty with responsiveness and adaptability. When goals and solutions are unclear and there is high volatility, there is particular need for alternative approaches to managing projects. Becoming equipped with different approaches to project management will allow project managers to better match the characteristics of the project at hand. The effort to accommodate agile project management approaches and learn how to be flexible and adaptable may well be worth the investment for many project managers. This flexibility could be highly advantageous when faced with certain types of projects and project scenarios".

On the other hand, the paper [11] wrote: "Although agile methodologies triumph over traditional ones in several aspects, there are many difficulties in making them work" and concluded the analysis without taking a position in favor of any Project Management methodology by writing "No matter what model is chosen for developing software applications, this activity involves complex processes that are often predisposed to errors". Moreover, the paper [12], after discussing the Waterfall and Agile Pros and Cons, concluded "it depends upon the organization which model to choose" and listed the two possible scenarios:

- "If requirement changes frequently and smaller projects, deliver product in short period time with skilled resources then we can choose Agile model;
- If requirement is clear, larger project then we choose Waterfall model"

Finally, we can conclude that there is not a clear position on which Project Management, between Waterfall and Agile, is the best. However, the aforementioned papers based their conclusion on empirical analysis taking into consideration only qualitative indicators and not any quantitative aspects. Therefore, the next paragraph will introduce some Key Performance Indicators (KPIs) aiming at differentiating the Agile methodology from the Waterfall technique by means of quantitative indexes.

2.4.2 Agile versus Waterfall: The Qualitative Delta

First of all, we take into consideration that Agile Project Management correctly works in framework characterized by a very low understanding of the systems in scope and by a very low certainty of outcome, or a medium combination of these two features. This result is due to the Agile flexibility and inclination toward changing environment, where the scope and the requirements can quickly vary because of the market or other internal and external factor. Instead, when the project concerns a high knowledge of the system framework and confidence on the information about Business Requirements and expected outcome, Waterfall has an effective Project Management and therefore, in such environment, it results a better approach with respect to Agile. The chart shown in Figure 14 give a graphic representation of this statement.

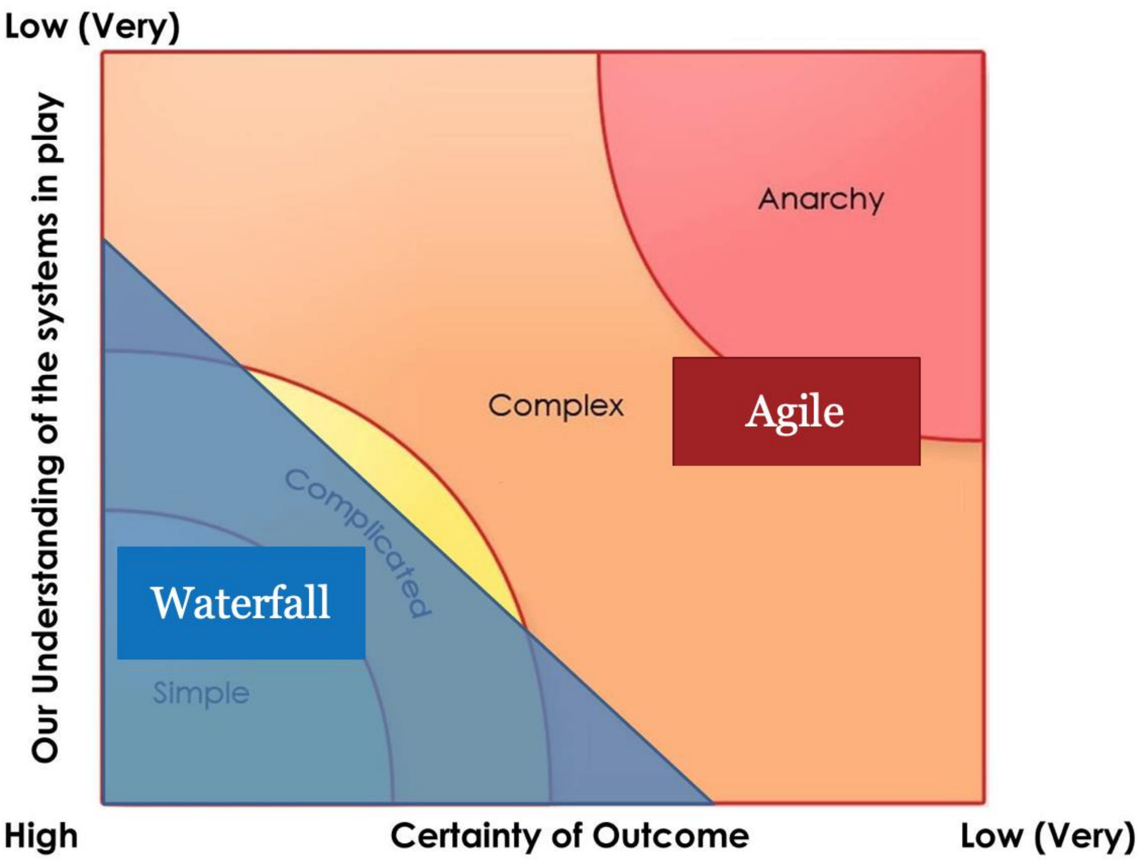

***Figure 14**: The different framework of Agile application with respect to the Waterfall*

Secondly, from a high point of view, the iterative nature of Agile methodology may be considered a competitive advantage with respect to the linear and sequential structure of the Waterfall approach. On one hand, Agile Project Management is value-driven, therefore the time and the cost of the project are fixed meanwhile the scope can change over time. On the other hand, Waterfall PM is plan-driven, meaning variable time and cost, and fixed scope (Figure 14).

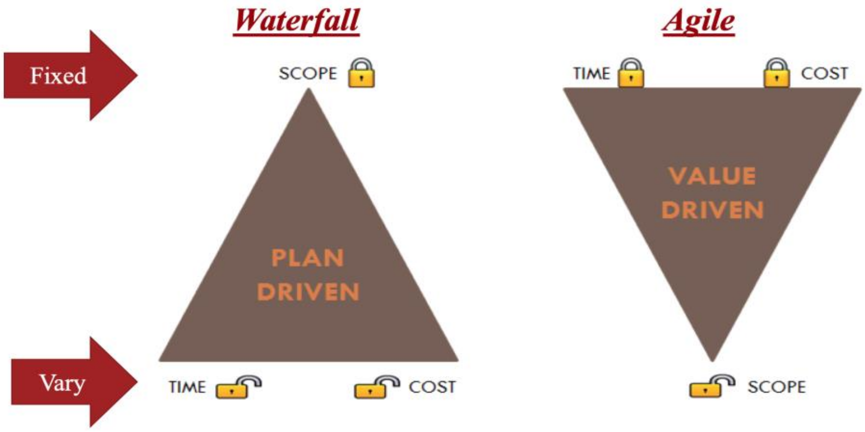

Figure 15: *The big picture of Waterfall versus Agile Project Management*

In addition to this analysis, the following principles identify the delta between the Agile approach and the Waterfall methodology.

- According to Highsmith and Cockburn [13] "what is new about agile methods is not the practices they use, but their recognition of people as the primary drivers of project success". Because Agile methodologies are **people oriented** and consider the whole team, from the developers or managers to the users or customers, as a key element in software development process. A clear example is given by the accurate definition of each role in Scrum Agile and by the purpose the Scrum Master has got with respect to the Development Team;
- Since software development processes cannot be accurately predicted far into the future, because of the elevate number of variables to consider, Agile can effectively **balance flexibility and planning** due to its iterative structure that results in a competitive advantage with respect to Traditional Project Management;
- **Simplicity** is another key success factor of Agile Project Management. Agile teams always take the simplest path consistent with their goals aiming at never producing more than what is necessary and never providing documents attempting to predict the future as documents will become outdated [14]. On the contrary, the Waterfall method aims at providing in its first phase a substantive document in support of the Business Requirements definition;

- Agile approach concerns a constant and frequent **User feedback** that encourages collaboration and discussion resulting in issue identification and problem solving. The User feedback in Waterfall is given only once, at the end of the project by means of the User Acceptance Test.

2.4.3 Agile versus Waterfall: The Quantitative KPIs

In order to perform an in-depth analysis that provides a quantitative approach to estimate the differences between Agile and Waterfall aiming at choosing the best Project Management strategy, the following Key Performance Indicators (KPIs) are investigated:

- Agile Project Management provides a greater **Return on Investment** (ROI) with incremental return. This is a consequence of the iterative structure of the Scrum Agile that concerns many potential shippable product increments during the whole duration of the project, rather than a single final delivery as it happens in Waterfall Project Management. Moreover, Agile approach provides quicker delivery and improved prioritization of high value items because requirements are continuously being re-evaluated, resulting in a faster time to market. This KPI is understood in relation to a fair and equal starting investment on both the Project Management methodologies (Figure 16). The measure criteria are the percentages: a result of 100% means that the project totally met the satisfaction and the requirements of the client, whereas 25%, 50% or 75% means that it made (more or less) partially and 0% the project does not return any advantages to the client;

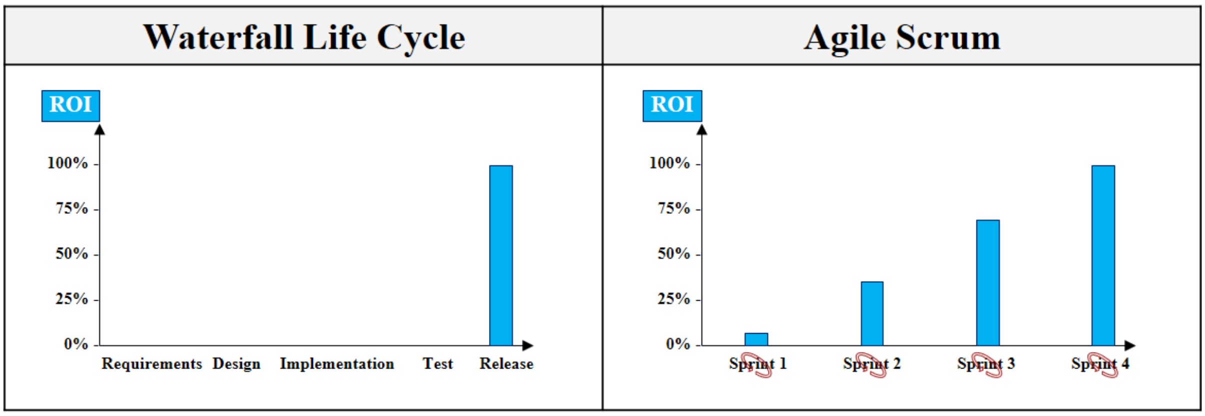

Figure 16: *Waterfall vs Agile – ROI*

- **Adaptability** with the environment, with the market or even with the User Business Requirements is a key benefit of Agile technique. Agile processes harness change for business' advantage with incremental software delivery whereas Waterfall methodology allows modification or integration of Business Requirements only during the first two phases: Requirements and Design; because any subsequent revision strongly increases both cost and time of the project. The measure criterion is a percentage scale where 100% means the project manages the environment changes, whereas 10% or lower means the project cannot handle any environment change;

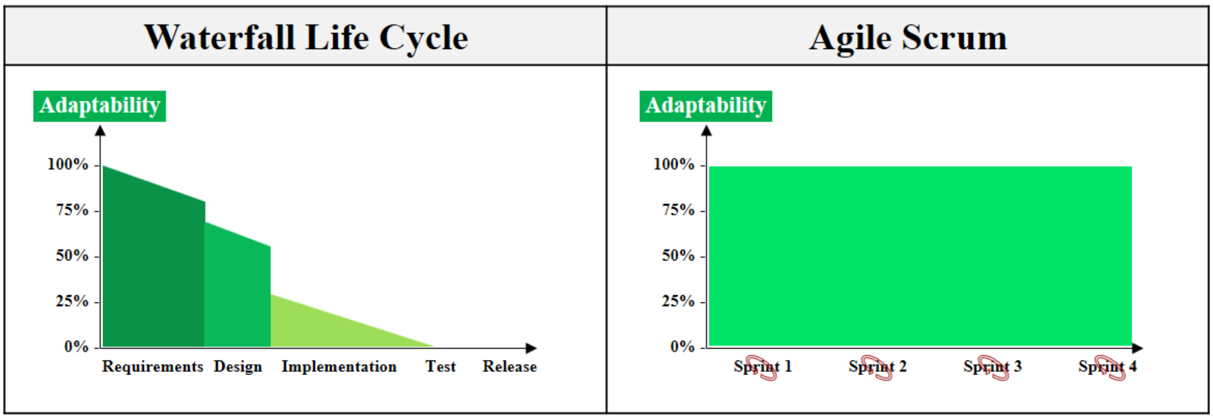

Figure 17: *Waterfall vs Agile – Adaptability*

- The Agile Scrum framework provides greater **visibility** through short release cycles, frequent demos of working software, and continuous business involvement. On the contrary, in Waterfall framework a sufficient visibility is given or at the very beginning or after the whole implementation phase when the final testing is running. The measure criterion is a scale for 1 to 5 points where 5 points mean the project manager has a perfect knowledge of the activities' status and their goodness, whereas 1 point or lower means the project manager has no idea about the activities' status and their goodness;

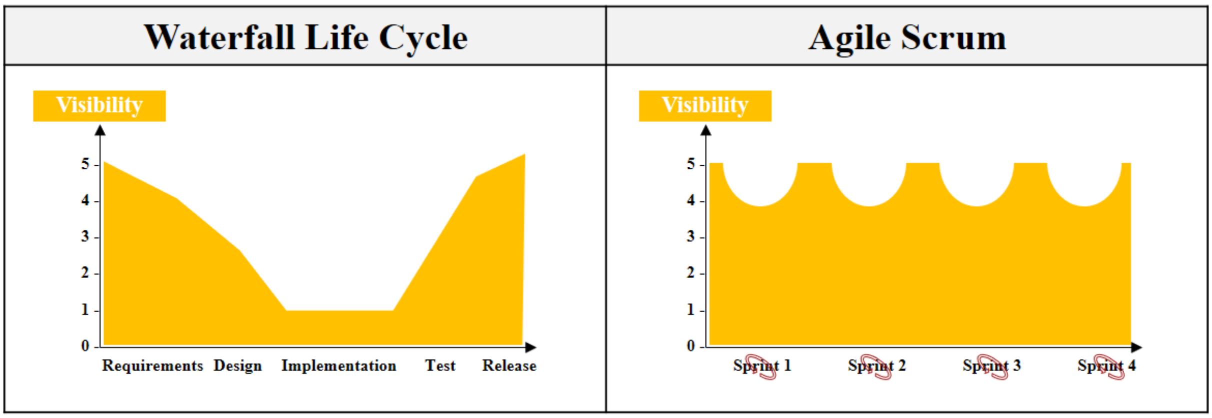

Figure 18: *Waterfall vs Agile – Visibility*

- **Work cost** is considered in term of how the project's cost is allocated in the timeline. Agile, as shown in Figure 15, has fixed cost because it provides a cross-functional team who works full time for the whole duration of the project. Waterfall provides unfixed cost that depends on the focus of the projects and that changes according to the operating phases. The measure criteria are the money (Euro) estimated in relation with a standard rate of effort cost;

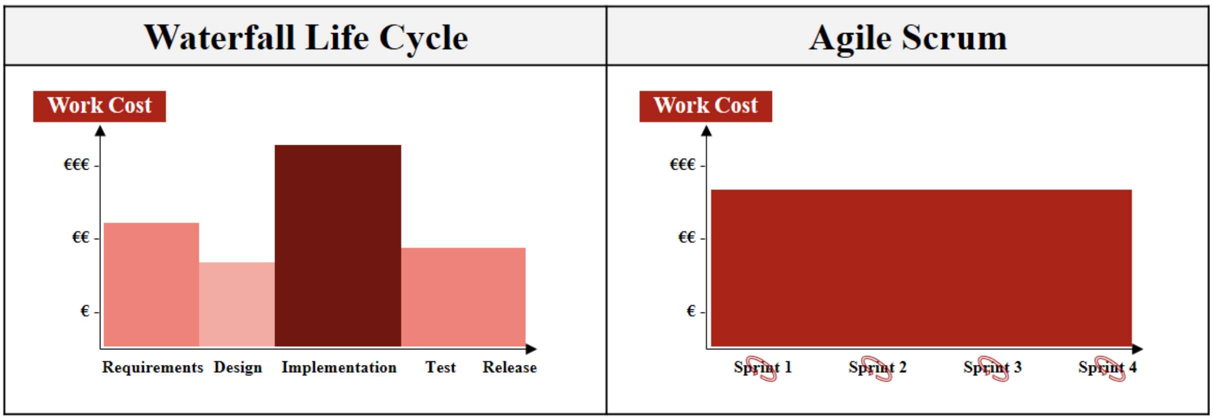

Figure 19: *Waterfall vs Agile - Work Cost*

- Agile Project Management improves overall quality through early defect detection and fixes leading to faster delivery of software to market resulting in low and steady **rework cost**. Waterfall Project Management instead presents a rework cost's trend that increases as the project comes to the end. As a consequence, software actions are usually not allowed in the final phase of Waterfall plan, but a new specific project is planned and developed. The measure criteria are the money (Euro) estimated in relation with a standard rate of effort cost;

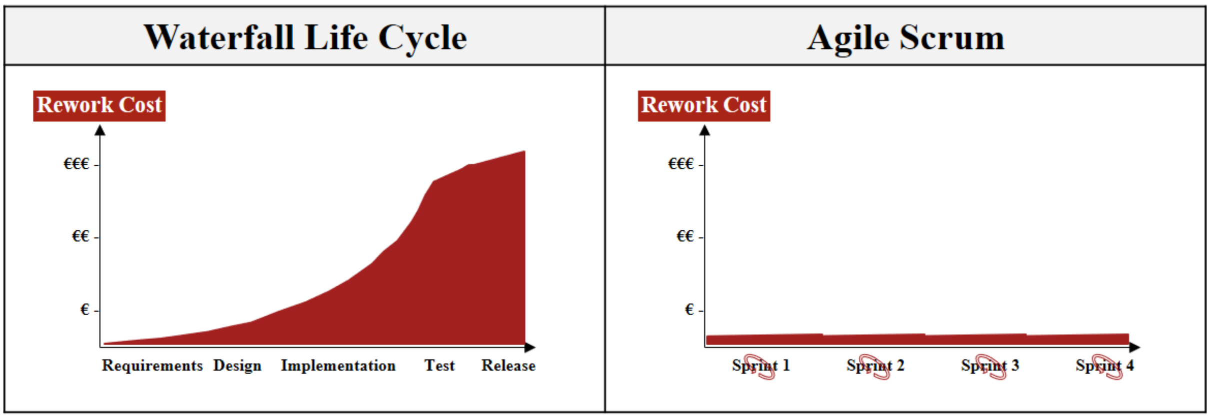

Figure 20: *Waterfall vs Agile - Rework Cost*

3. Case Study of the IT Banking System

The following chapter concerns the case study of the Bank. First, we will introduce the Information Technology background of the Bank by describing and showing its IT structure and its dataflows from the native data to the processed and reported output data used by the Business Units. Then, we will illustrate the project portfolio that includes the different typologies of activities, which are under analysis, and the respective operative boundary of the IT architecture and data streams. The Project Management actually applied by the Bank will therefore in-depth analyzed aiming at identifying its benefits and its drawbacks thanks to the theoretical study developed in chapter 2.

The understanding of such background is essential for achieving the goal of the thesis: to enhance the traditional Project Management framework. We will indeed analyze when and how the Agile approach can improve the management of the activities as well as the performance of the IT structure. This process will include hypothetical examples in support of the thesis.

For compliance reasons, all the names, dates, numbers indicated in the following case study are invented in order to mask any kind of reference to a real person, place, structure or institution as well as the Bank itself.

3.1 Banking Information Technology Framework

Aiming at simplifying the analysis, we will assume that the 'Bank' is composed by two different departments: the 'Business Unit' (BU) represented by the 'User' as well as the 'Client', and the 'IT Unit' (IT) that will be described in detail. The BU needs for computing economic data for financial reasons (risk analysis, accounting balance, regulatory purpose, business strategy, internal policy) and, therefore, needs for an effective software and a solid hardware. The IT is responsible for the governance of the software and hardware of the bank aiming at controlling the maintenance of the Information Technology system, at regularly developing new software functionality and at updating the hardware.

The framework of the case study concerns the Software Development Life Cycle (SDLC) the IT implements in order to meet the Business Requirements expressed by the BU. The BU is therefore the Business Owner who requests for specific software developments, pays the implementations and takes benefits from them.

The Bank has many IT divisions as well as BU sectors and generally each BU has a dedicated IT with a definite structure. However, there is a lot of synergy between all the departments and many projects are cross between them; more the number of iterations, higher the complexity of the activities. To better understand how the iterations between the ITs come, we will first introduce the standard IT structure.

3.1.1 The Architecture of the IT Unit

First, we introduce a standard architecture of the ITs structures that aims at receiving input data, elaborating such data and reporting the output data or feeding another IT. Then, we will extrapolate the architecture of the IT object of the analysis.

We can describe the dataflow through the IT-example structure as follow:

- The **Source Systems** are some IT Units external to IT-example that works as data feeding toward IT-example. The Source Systems can be the real sources of the native data (i.e. a branch of the Bank) or they can receive the native data by other Source Systems and let them pass through their architectures without elaborating it as well as after a dedicated data processing. A single Source System can feed many IT Units in different ways depending on the requirements of the data itself;
- The **Pre-Processing** is the first part of IT-example. The Pre-Processing collects all the data fed by its many Source Systems aiming at organizing all the data received and feeding the following systems. IT-example can have more than one Pre-Processing system, depending on the typology of data it collects, for example domestic data and data concerning foreign banks may be fed by different Source Systems in different Pre-Processing systems;
- The **Processing Engine** is the heart of IT-example as it is where the data processing is performed. The data feeding generally comes by a single Pre-Processing system, therefore if IT-example has more than one Pre-Processing system or it has the same number of Processing Engines or the different Pre-Processing systems are sequential and merge into a single data feeding toward the Processing Engine. As the data has been processed, the output can be sent toward many possible directions: to a reporting process (described below), to other IT Units, to another Processing Engine or, moreover, to a combination of them. The latter case means that IT-example can have a system that works as both Processing Engine for data processing and Pre-Processing as data feeding;

- Finally, IT-examples has the **Post-Processing** system that is a reporting process aimed at making the output of the data processing visible to the Users of the BU-example. IT-example can have different tools of reporting. Since the recording should be user friendly, BU-example generally decides how to manage the view of the output data.

In order to help the understanding of the description above, two different examples of IT architecture are graphically reported: Figure 21 shows the simplest IT case whereas Figure 22 shows a complex IT situation.

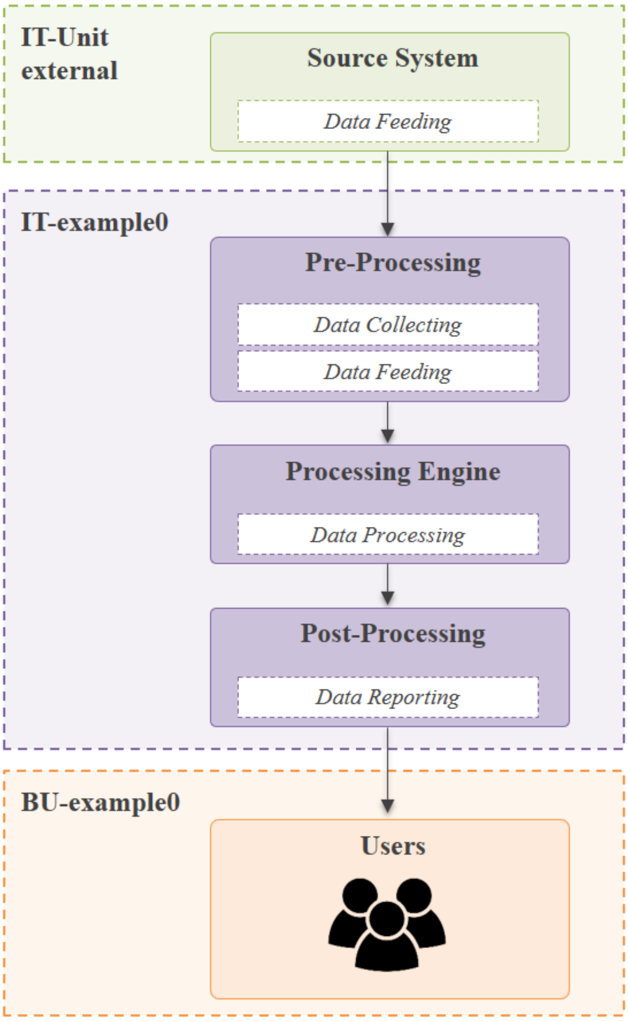

Figure 21: *Simple IT architecture's example*

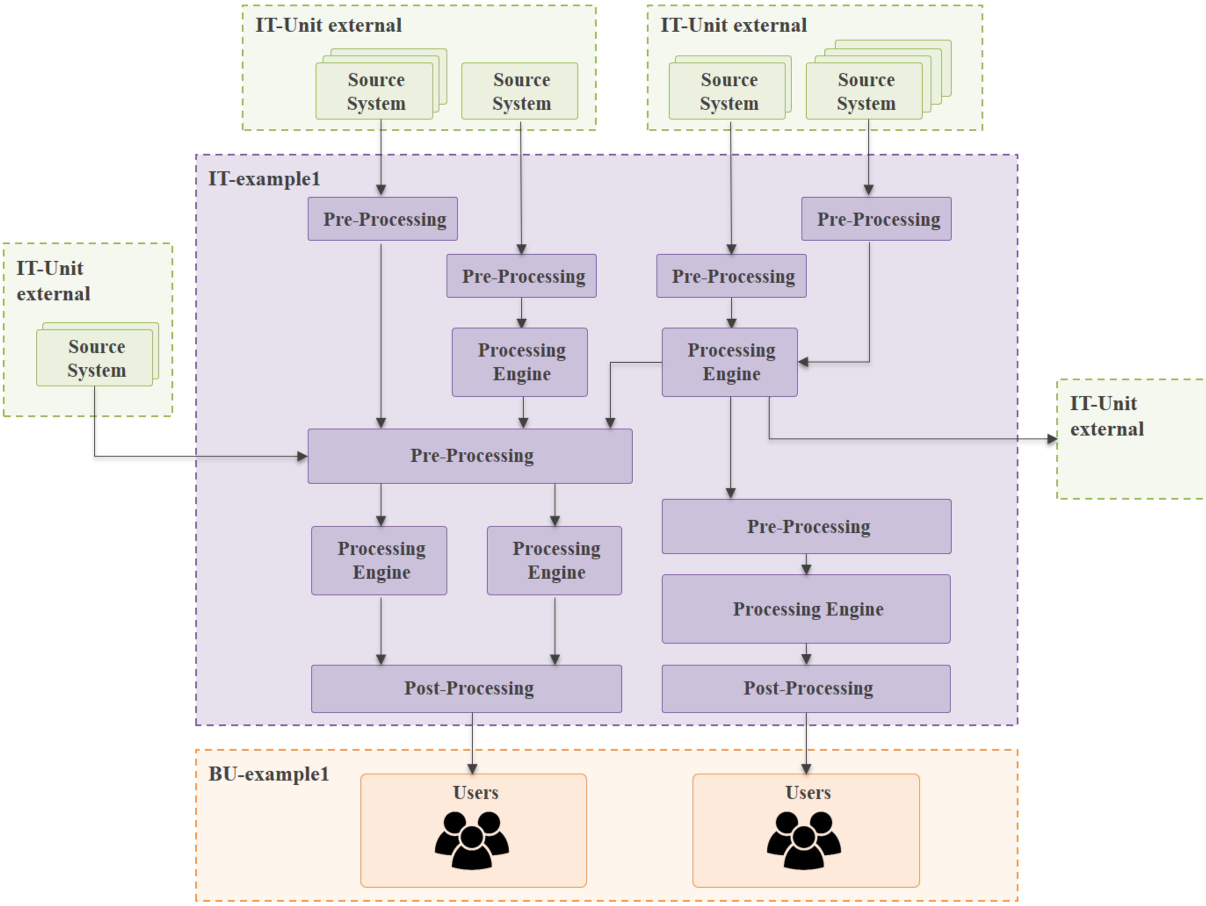

Figure 22: Complex IT architecture's example

3.1.2 The structure of the IT-Alfa architecture

Since we have introduced the general concepts of the IT architectures, in this paragraph we will describe the IT structure of the IT Unit object of analysis, called IT-Alfa.

IT-Alfa consists of the following systems:

- **Alfa1** is the upstream application of IT-Alfa and it is fed by many Source Systems with different frequencies. Alfa1 is indeed a Pre-Processing system that applies data collecting and data feeding. All the data collected in Alfa1 are sent to Alfa2 and to no one else;
- **Alfa2** is fed both internally by Alfa1 and externally by many Source Systems. Alfa2 works as Processing Engine as it applies may computation logics to the data collected and, after, it works as data feeding toward Alfa 3 and toward other external IT units called IT-Delta and IT-Epsilon. IT-Delta concerns only a Post-Processing system which governance is out-of-scope with respect to IT-Alfa even if some Users of BU-Alfa

actively use IT-Delta. IT-Epsilon is another IT Unit self-contained that use Alfa2 as a Source System;

- Alfa3 works as both data collecting and data feeding and it is the most complex system of the architecture. It is indeed shared by three different IT Units because it is divided in three subsystems: **Alfa3A** concerning the section related to IT-Alfa, **Alfa3B** concerning the part related to IT-Beta and **Alfa3C** concerning the section related to IT-Gamma. These three subsystems share the same DataStage, which is the tool that can extract data, transform it, apply business principles and then load it to any specific target, but they have different dataflow. Details follow below:
  - Alfa3A is fed mainly by Alfa2, but also by Alfa3B about a limited operating area; it feeds Alfa3C and Alfa4A;
  - Alfa3B is fed mainly by the Source Systems related with IT-Beta, but also by Alfa2 about a significant amount of data; it feeds Alfa3A and Alfa4B;
  - Alfa3C is fed only by Alfa3A; it feeds Alfa4C;
- Alfa4, as it has been revealed on the bullet point above, is divided in three subsystems as well as Alfa3. However, **Alfa4A** for IT-Alfa, **Alfa4B** for IT-Beta and **Alfa4C** for IT-Gamma do not share any hardware or software because they are independent between them. They represent the Processing Engines of the respective IT Units that works separately but actually with almost the same computation logics that depends on the requirements expressed by the respective Business Units. Details follow below:
  - Alfa4A is fed by Alfa3A and meets the Business Requirements of BU-Alfa; it sends the output of the Engine to Alfa5;
  - Alfa4B is fed by Alfa3B and meets the Business Requirements of BU-Beta; it makes the output data visible to BU-Beta with a dedicated reporting process;
  - Alfa4C is fed by Alfa3C and meets the Business Requirements of BU-Gamma; it makes the output data visible to BU-Gamma with a dedicated reporting process;
- **Alfa5** is the downstream application as the Post-Processing system of IT-Alfa and it provides the reporting procedure aimed at recording the engine output to BU-Alfa.

In order to better understand the complex scheme of the IT-Alfa architecture, Figure 23 shows its dataflow logic and structure.

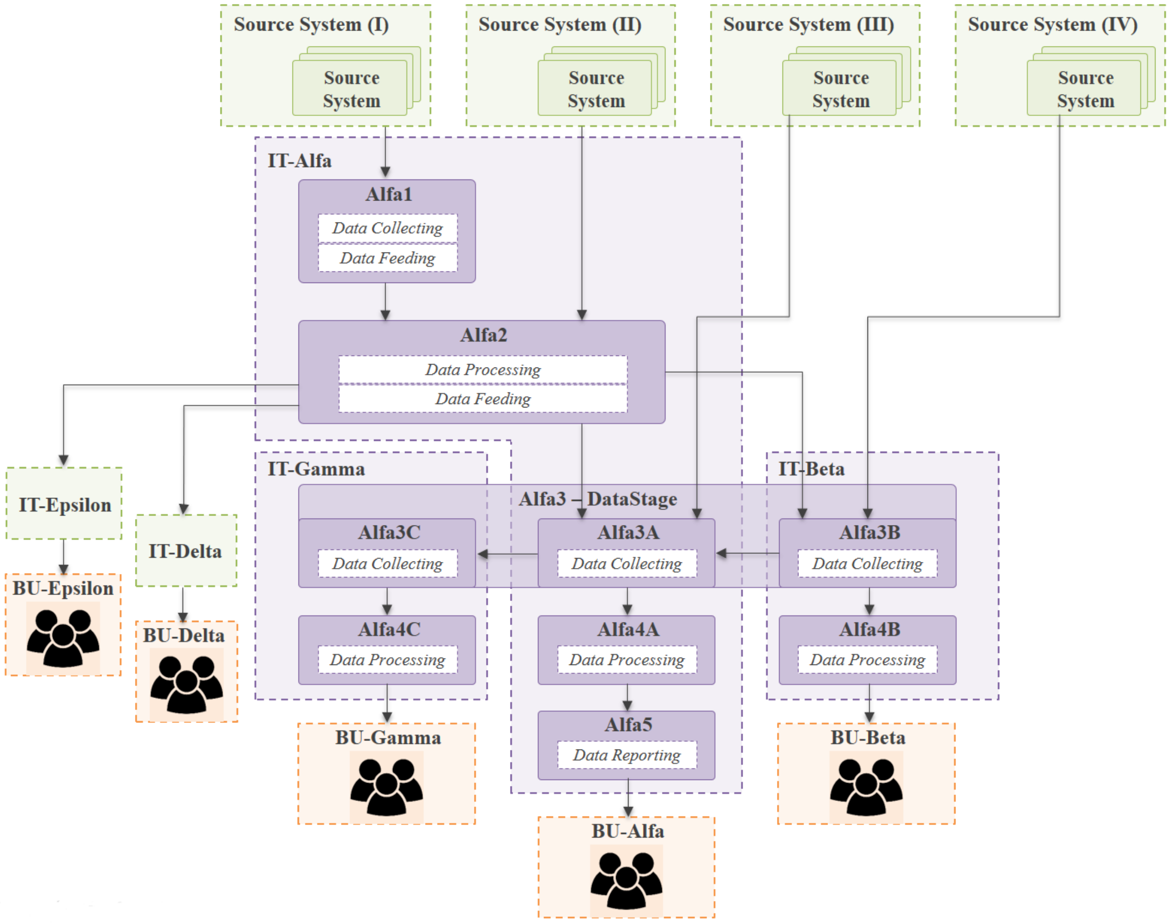

Figure 23: *IT-Alfa Architecture*

3.1.3 The IT-Alfa Dataflows

The complex IT-Alfa structure concerns different dataflows that depend on where the data comes from (Source System) and where the data is sent to (BU-Unit). Below, we list all the possible combinations of Dataflow that IT-Alfa has:

- **IT-Alfa_Dataflow1:** *Source System (I) → Alfa1 → Alfa2 → IT-Delta → BU-Alfa/ BU-Delta;*
- **IT-Alfa_Dataflow2:** *Source System (I) → Alfa1 → Alfa2 → IT-Epsilon → BU-Epsilon;*
- **IT-Alfa_Dataflow3:** *Source System (I) → Alfa1 → Alfa2 → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow4:** *Source System (I) → Alfa1 → Alfa2 → Alfa3A → Alfa3C → Alfa4C → BU-Gamma;*

- **IT-Alfa_Dataflow5:** *Source System (II) → Alfa2 → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow6:** *Source System (II) → Alfa2 → Alfa3A → Alfa3C → Alfa4C → BU-Gamma;*
- **IT-Alfa_Dataflow7:** *Source System (II) → Alfa2 → Alfa3B → Alfa4B → BU-Beta;*
- **IT-Alfa_Dataflow8:** *Source System (III) → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow9:** *Source System (III) → Alfa3A → Alfa3C → Alfa4C → BU-Gamma;*
- **IT-Alfa_Dataflow10:** *Source System (IV) → Alfa3B → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow11:** *Source System (IV) → Alfa3B → Alfa3A → Alfa3C → BU-Gamma.*

The following figures provide a graphic interpretation of each of the eleven possible dataflow.

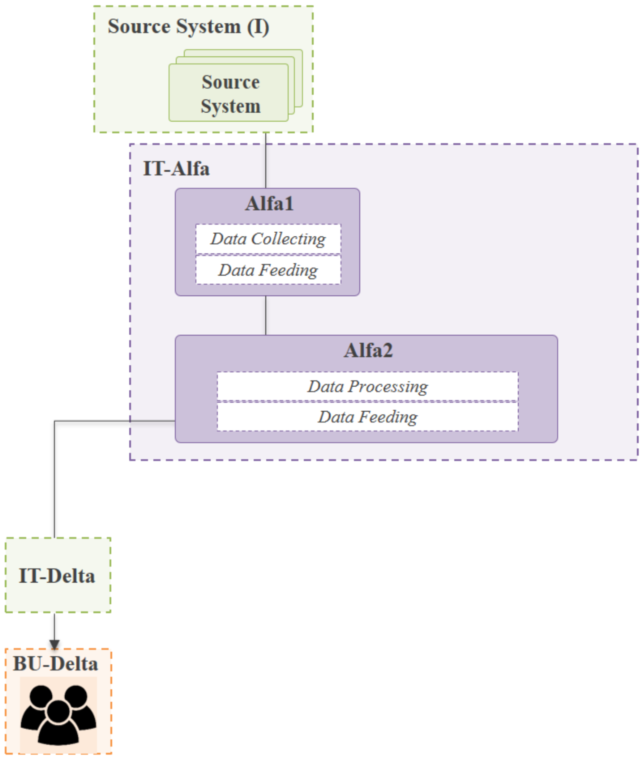

Figure 24: *IT_Alfa-Dataflow1*

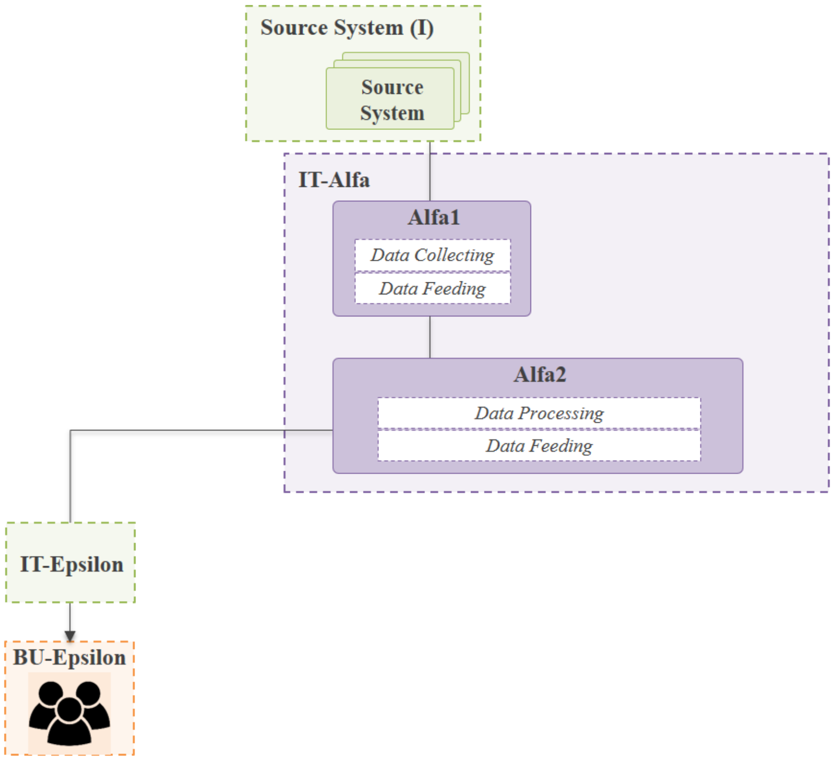

Figure 25: *IT-Alfa_Dataflow2*

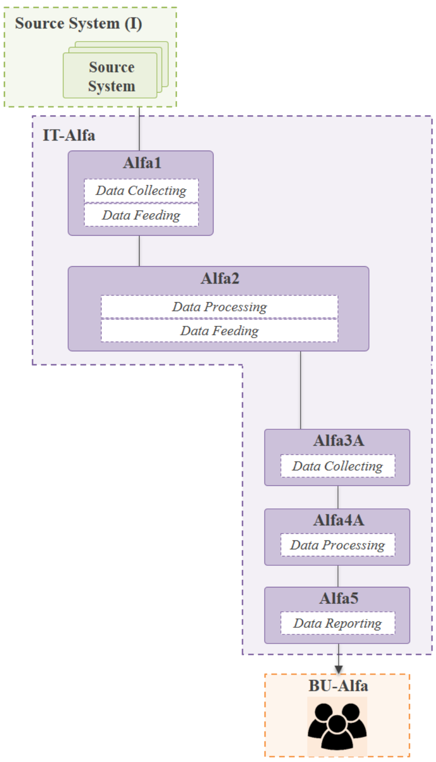

Figure 26*: IT-Alfa_Dataflow3*

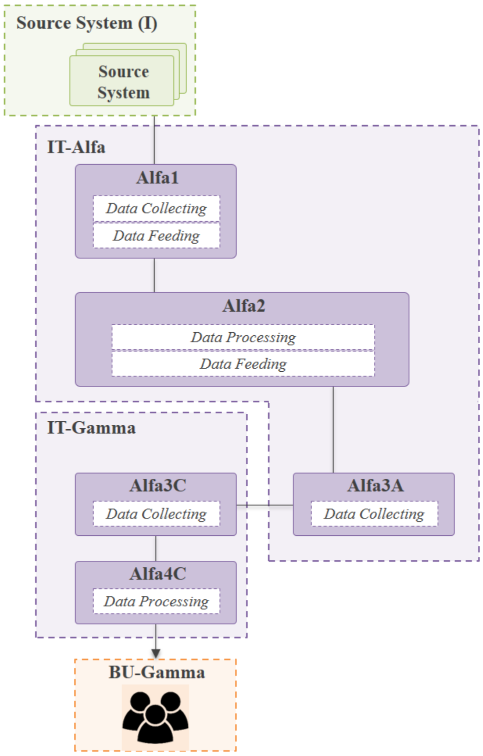

Figure 27: *IT-Alfa_Dataflow4*

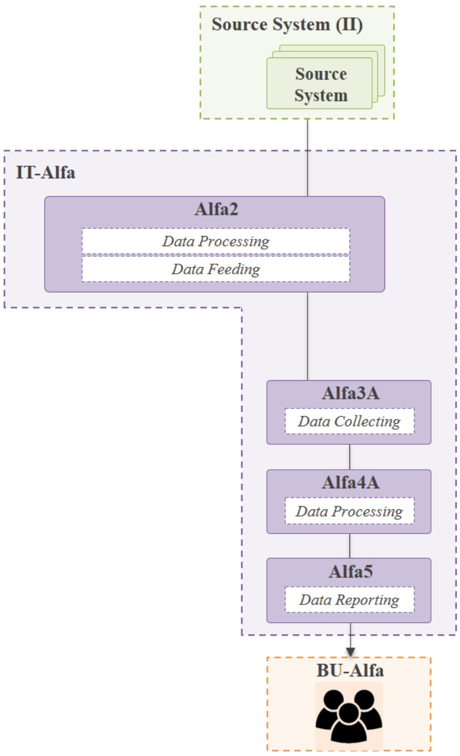

Figure 28: *IT-Alfa_Dataflow5*

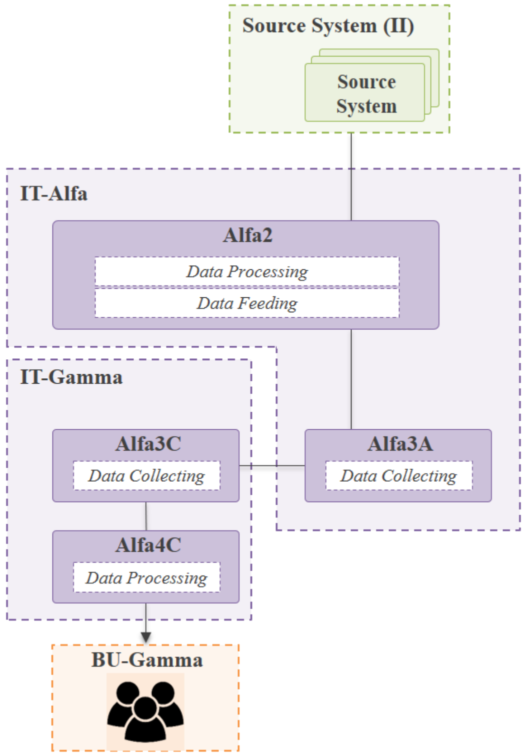

Figure 29: *IT-Alfa_Dataflow6*

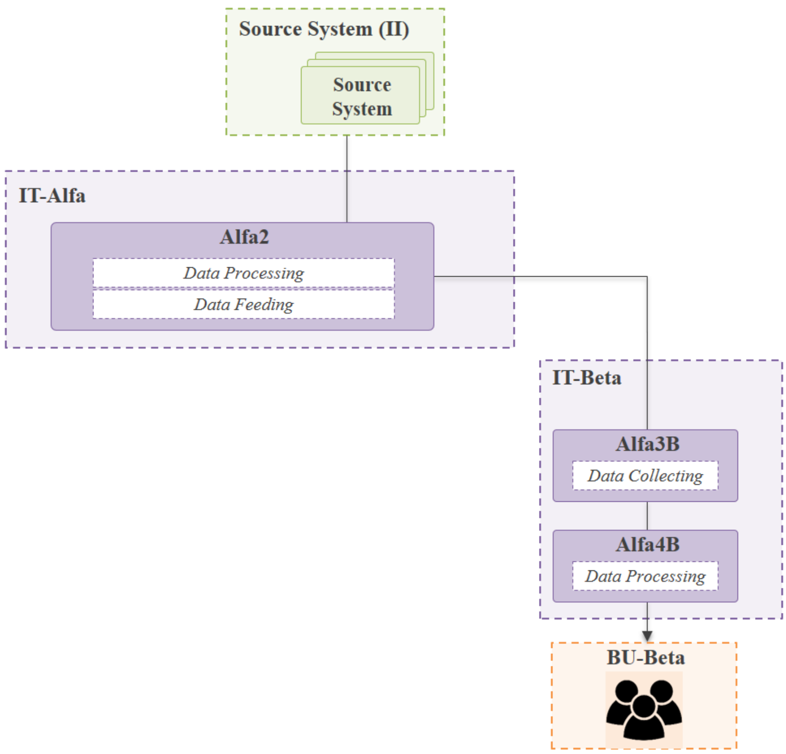

Figure 30*: IT-Alfa_Dataflow7*

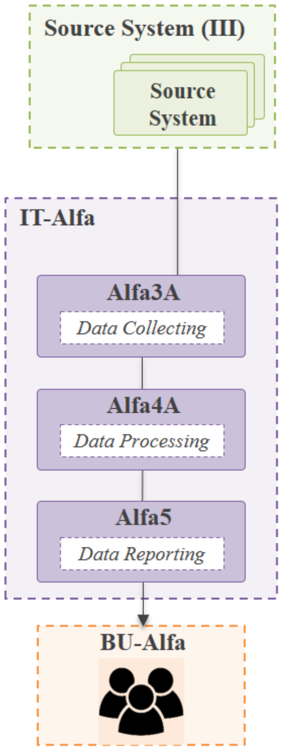

Figure 31: *IT_Alfa-Dataflow8*

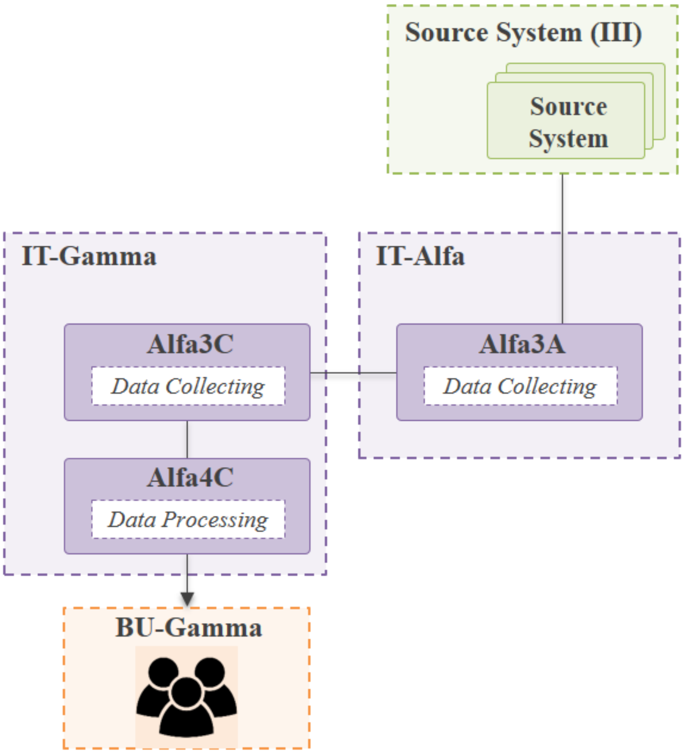

Figure 32: *IT_Alfa-Dataflow9*

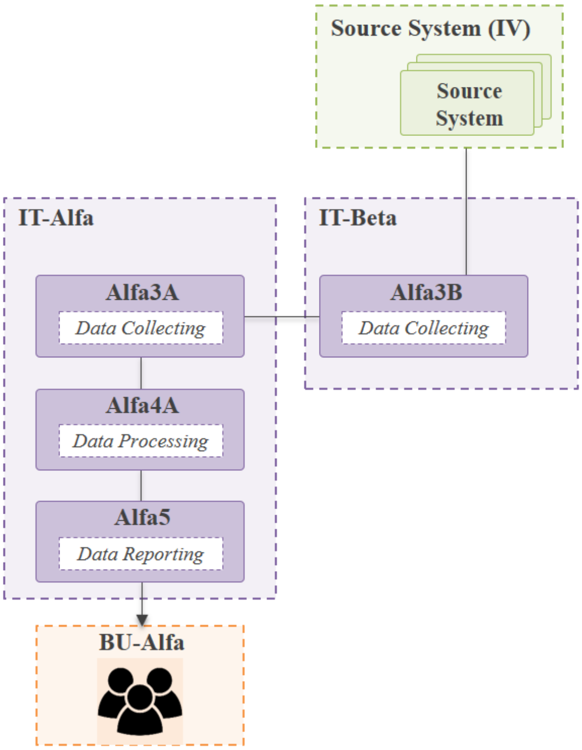

Figure 33: IT_Alfa-Dataflow10

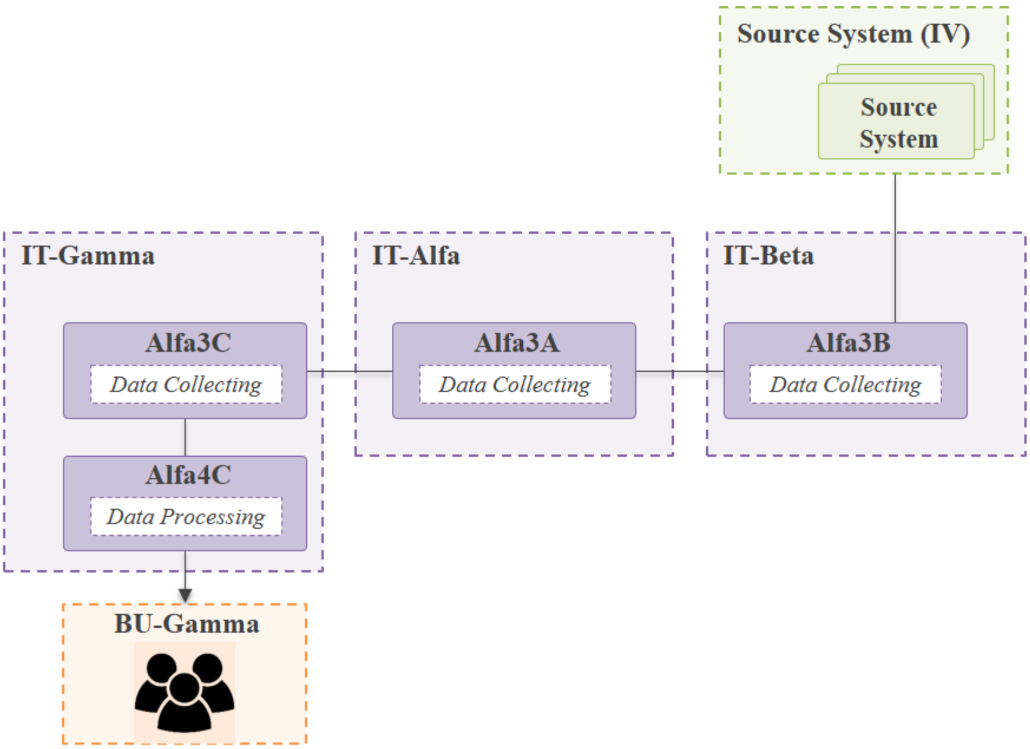

Figure 34: IT_Alfa-Dataflow11

3.2 Project Portfolio and Activity Scope

Since we have previously introduced the IT architecture of the Bank, this section will concern the different typologies of projects developed by IT-Alfa. A typical Bank's project portfolio is huge, and each project is characterized by many different features. Therefore, we will structure the analysis by taking into consideration four macro areas of activities related with IT-Alfa: the BU-Alfa ventures, the IT-Alfa contributions to external BUs, the technological upgrade of IT-Alfa and the ordinary activities required by the Bank institution.

3.2.1 The BU-Alfa Ventures

BU-Alfa plays a specific economic role for the Bank because, for example, BU-Alfa is responsible for the investment strategy of the Bank or for the market risk analysis as well as the internal risk management. To accomplish its investigation, examination and evaluation, BU-Alfa requires to IT-Alfa the governance of the IT structure that makes all these computations possible. Since the European Central Bank (ECB) as well as the European Banking Authority (EBA) systematically issued, and are currently issuing, a lot of laws and guidelines for the governance of the Bank, BU-Alfa decides every year the most relevant projects that IT-Alfa must implement, aiming at respecting the past and future regulations. This type of project is the BU-Alfa venture that has proper meaning for the BU-Alfa work and it is financed by BU-Alfa.

The BU-Alfa ventures are usually defined at the beginning of each year by taking into consideration the regulatory scope and the available capital budget, however, new necessities may occur during the whole year and, therefore, BU-Alfa may require new activities. The aims of the project can be various as the introduction of a new functionality, the improvement or change of an existing feature, the integration of new dataflow or information, the addition of a different feeding frequency.

BU-Alfa is the Business Owner of this type of activity as the BU-Alfa users write the Business Requirements, by also helping IT-Alfa understanding them, and pay the effort needed, as well as the necessary hardware, for implementing the solution that meets such Business Requirements.

By looking at the IT features, the BU-Alfa ventures typically require new, or improved, computation logics, meaning that the implementation activity focuses on the data processing and therefore on the Processing Engine. Because IT-Alfa, as we have seen in paragraph 3.1.2, has more than one Processing Engine, the IT structure in scope of the project may change.

Below we list the four possible dataflows concerning the BU-Alfa ventures that are graphically shown in paragraph 3.1.3 (cfr. Figure 24, Figure 26, Figure 28, Figure 31, and Figure 33):

- **IT-Alfa_Dataflow1:** *Source System (I) → Alfa1 → Alfa2 → IT-Delta → BU-Alfa/ BU-Delta;*
- **IT-Alfa_Dataflow3:** *Source System (I) → Alfa1 → Alfa2 → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow5:** *Source System (II) → Alfa2 → Alfa3A → Alfa4A → Alfa5 → BU-Alfa;*
- **IT-Alfa_Dataflow8:** *Source System (III) → Alfa3A → Alfa4A → Alfa5 → BU-Alfa.*
- **IT-Alfa_Dataflow10:** *Source System (IV) → Alfa3B → Alfa3A → Alfa4A → Alfa5 → BU-Alfa.*

Because, as one may expect, the final output always goes to BU-Alfa, the discernment for determining the correct Dataflow is the Source System from where the input data in scope of the project comes.

IT-Alfa should prioritize the BU-Alfa ventures over other activities in order to meet the desiderata expressed by the BU-Alfa users.

3.2.2 The IT-Alfa Contributions to External BUs

As we have seen in paragraph 3.1.2, the IT-Alfa architecture concerns data feeding, data collecting, data processing and data reporting not only for BU-Alfa but also for others Business Units as BU-Beta, BU-Gamma, BU-Delta and BU-Epsilon. All these Business Units have the same duty and role in the Bank as the ones explained in the section above for BU-Alfa, but with slightly or completely different goals. It means that each BU decides a definite number of activities for the year that must be implemented not only by the IT department responsible for the BU that requires the project, but also by any other IT units that apply any kind of data management inside the dataflow in scope.

Such kind of project is the contribution that IT-Alfa provides for all the Business Units different from BU-Alfa. Moreover, the IT-Alfa contributions also concern all the activities performed by the Source Systems that have impacts on IT-Alfa.

In both the cases, even if the Business Owner, that finances the project, is the BU which requires the new product or the new functionality, the Business Requirements for IT-Alfa are written by

BU-Alfa. There is indeed a dedicated communication channel between the different Business Units aimed at defining the different impacts on each department of the Bank and the resulting Business Requirements.

The IT-Alfa contributions may concern all the eleven different IT-Alfa Dataflows listed in paragraph 3.1.3. However, the weight of the impacts strongly changes between the various streams of data and definitely depends on the project scope.

3.2.3 The Technological Upgrade of IT-Alfa

As a consequence of the introduction of new functionalities in the IT system and new information in the dataset due to the previously mentioned projects, it is always more difficult for IT-Alfa to guarantee the correct performance of the software and the hardware. Therefore, IT-Alfa provides specific activities aimed at strengthening the IT structure and at increasing, or at least maintaining, the hardware performances.

The hardware upgrades are complex in term of organization and requirements and there is a specific department, named HW_Team, that presides over all the machines and physical application. IT-Alfa has old application systems and it may take almost one year for achieving an upgrade. Moreover, there is not a clear process for the definition of the requirements because, on one hand, IT-Alfa suppose a possible solution, on the other hand, HW_Team expresses their point of view.

This kind of activity may concern the whole IT-Alfa structure, many different IT units or only few applications and, as a consequence, the dataflow in scope varies according to the project.

3.2.4 The Bank Ordinary Reorganization

In addition to the projects previously described, the Bank institute regularly requires activities of internal reorganization. The Business Owner of this kind of projects are specific work sites that operate across the whole Bank. The impacts on the IT departments may be different and for every activity there is a specific feasibility study.

Even if the requirements are typically very clear from the beginning, the projects imposed by the Bank may be critical due to the high number of actors involved, that means no possibility of delay on the schedule, and because the requirements are imposed, and no changes or exception are admitted.

There are no contributions between the IT units; every departments estimate their respective impacts and communicate the effort required with standard procedures. The projects do not concern a definite dataflow but rather a specific upgrade for each application of the IT unit.

3.3 IT-Alfa Waterfall Project Management

This section aims at reporting how Project Management is developed in IT-Alfa. Since the banking IT framework has been described in paragraph 3.1 and the project portfolio has been investigated in paragraph 3.2, the Project Management approach will be explained for each of the possible type of project with reference to the IT-Alfa architecture.

As we will see, IT-Alfa applies a traditional Waterfall approach for managing the projects, but, in order to explain the picture As-Is of the IT governance on the software development, we will first introduce the main actors involved in the activities and then we will deep dive into the management dynamics.

3.3.1 Main Actors and their Roles

This section will specify the full list of the actors that plays a given role in the Project Management lifecycle of IT-Alfa. Below the details:

- **IT-Alfa_Manager:** He or she is the Project Manager as well as the responsible of IT-Alfa. IT-Alfa_Manager perfectly knows the IT-Alfa architecture and takes all the final decisions. He or she is a Bank employee. This position and its description are valid for all other IT units;
- **IT-Alfa_Analyst:** He or she is the technical expert of IT-Alfa. IT-Alfa_Analyst checks and validates all the technical and technological aspects of IT-Alfa. He or she can be a Bank employee or a consultant. Other IT units requires this position as well;
- **IT-Alfa_PMO:** He or she provides methodological and operative support to IT-Alfa for effectively managing the projects and meeting the business requirements. IT-Alfa_PMO usually is a consultant but can also be a Bank employee. Not any IT unit has a PMO and some PMO may follow more than one single IT unit;
- **IT-Alfa_Team** represents the ensemble of IT-Alfa_Manager, IT-Alfa_Analyst and IT-Alfa_PMO;
- **Alfa#_Team:** Any application system Alfa# (cfr. Paragraph 3.1.2) has a dedicated team usually composed by one manager and one or more developers/ analysts. In some cases,

the manager is also a developer or an analyst. The team members are typically consultants, but they can either be only Bank employees or a mix.

- **BU-Alfa_User:** He or she is the final user of the IT service provided by IT-Alfa. BU-Alfa_User is a Bank employee. This position and its description are valid for all other Business Units;
- **Alfa_ServiceManager:** He or she manages the technical aspect of the budget and the payments. Alfa_ServiceManager is also a formal position of communication between IT-Alfa and BU-Alfa. He or she is a Bank employee.

3.3.2 BU-Alfa Ventures Planning

At the beginning of the year, BU-Alfa decides the most relevant economic themes related with its financial functionality inside the Bank and, therefore, establishes the new projects to carry out. After this business settlement, the BU-Alfa office, composed by BU-Alfa_Users and their director, communicates with the help of Alfa_ServiceManager to the IT-Alfa office, composed by BU-Alfa_Manager and his/her director, the new future activities by providing an overview of the requirements and the goals.

After this kick-off meeting, BU-Alfa_Users write the Business Requirements for each activity and, as they formally delivery the Business Requirements Book (BRB) of a specific project, such project begins.

As in the Waterfall Project Management lifecycle described in paragraph 2.2.1, the first phase is the Feasibility Study based on the Business Requirements.

Let's assume that Activity1 is the project described by the BRB delivered by BU-Alfa at the beginning of the year. Then, IT-Alfa_Manager takes charge of the BRB and, with the collaboration of IT-Alfa_Analyst, examines the possible impacts of such requirements. First impact to analyze is the operative boundary, that means understanding which dataflow is in scope and which application systems requires a development. Since Activity1 requires the application of new computation logic in the Processing Engine Alfa4A, IT-Alfa_PMO schedules a meeting with Alfa4A_Team and IT-Alfa_Team. As they have in-depth analyzed the BRB and understood the requirements at the Processing Engine level, IT-Alfa PMO plans a meeting with all the application systems affected by Activity1 and, assuming that Activite1 concerns IT-Alfa_Dataflow8 (cfr. Figure 31), the meeting table will include Alfa3A_Team, Alfa4A_Team, Alfa5_Team and IT-Alfa_Team. Then, because the output of the aforementioned meeting is a set of solutions that needs for some business decisions or clarifications, IT-Alfa_PMO schedules a more formal conference with all the stakeholders of the projects: all the impacted IT teams, the BU-Alfa_Users and Alfa_ServiceManager. At some point, the meetings should also include the Source System (III) to ensure the correct data feeding toward Alfa3A.

This brainstorming cycle continues until the Business Requirements are definitely clear to all the stakeholders and in particular to the IT teams that need for the implementation. The time horizon this process takes to complete depends on the complexity of the project and on the number of actors involved. For example, assuming that Activity2 impacts IT-Alfa_Dataflow3 (cfr. Paragraph 3.1.3), the Feasibility Study of Activity1 will theoretically results quicker than the one of Activity2 since Activity1 concerns two less application systems.

In this phase, the documents play a significant role. On one hand, any business decision must be clearly written in the BRB and, if necessary, the BRB must be updated during the Feasibility Study. On the other hand, IT-Alfa_PMO collects all the technical requirements needed for meeting the BRB in a specific document that must be validated by both the technical and the business stakeholders.

As the Technical, Legal and Operation Feasibility (cfr. Paragraph 2.1.1) for Activity1 are successfully completed, Alfa3A_Team, Alfa4A_Team and Alfa5_Team estimates the effort needed for implementing Activity1 and IT-Alfa_Manager communicates the total effort cost to Alfa_ServiceManager who executes the administrative and bureaucratic tasks for obtaining the budget approval and concluding the Economic Feasibility.

Last step for concluding the feasibility of Activity1 is the development planning's definition. IT-Alfa_Manager decides when to collocate Activity1 by taking into consideration the other already planned projects, the priority levels and the BU-Alfa's desiderata for the delivery. Then, the operative plan is validated first with the technical teams (Alfa3_Team, Alfa4A_Team and Alfa5_Team) and second with BU-Alfa_Users for the final approval.

Since the planning structure mirrors the Waterfall methodology, the following phases are provided:

- **Design:** Alfa3A_Team, Alfa4A_Team and Alfa5_Team design the software implementation of Activity1 in relation with their respective application systems. Then, each team writes the development process Activity1 requires in two different documents: one for the functional analysis and another one for the technical analysis. The functional development design, after the IT-Alfa_Manager supervision, is delivered to the BU-Alfa_Users for obtaining its approval, whereas the technical analysis is directly validated by IT-Alfa_Analyst. The list of documents provided can change depending on the project impacts. Generally, it takes two weeks for the documents' drafting and one week for the document's validation.

- **Implementation:** Once every development design has been approved, the teams begin to implement the new software in their respective application systems. Because this phase, that typically is the longest, must guarantee the fulfillment of the requirements by transforming them on the specifications of software technologies, any error about technical or business requirements must immediately come up aiming at solving it quickly. Since IT-Alfa_Manager has a slight visibility about the progress of this phase, a proactive behavior is required from the development teams as well as a weekly monitoring by IT-Alfa_PMO is necessary. This phase typically includes some technical tests performed independently by each application system in scope aimed at guaranteeing the correct development of the software. Generally, the implementation phase takes from two to four weeks depending on the project complexity.
- **System Integration Test (SIT):** Since the software implementations are concluded, the Test phase can start. The testing process for Activity1 is divided into two parts: first a System Integration Test between all the application systems is carried out, second the output of the SIT is reported to BU-Alfa_Users for the User Acceptance Test (UAT) and the user certification. The tests have a common perimeter, in terms of data boundary and data cut-off, that is consolidated along with the planning. Therefore, Source System III feeds Alfa3A (step1) that collects all the input data (step2) and sends it toward Alfa4A (step3). As Alfa4A is fed, the data processing is performed (step4) and the output data is sent to Alfa5 (step5) that reports the results to BU-Alfa_Users (step6). Each of the aforementioned step has a specific timeline aimed at preventing any possible delay. Overall, the SIT takes about two weeks depending on the dataflow in scope: the more the application systems involved are, the longer the time to complete the test;
- **User Acceptance Test (UAT):** As the SIT output is reported to BU-Alfa, the BU-Alfa_Users analyze it. Then, there are two possible scenarios: or the output is correct and meets the requirements or the data analyzed shows errors and anomalies. In the latter case, the decisions to take can be many depending on the nature and the weight of the issue (we will focus on this topic in paragraph 3.3.3), whereas in the positive scenario

the software development is approved by BU-Alfa and can be released. Due to the high importance of the decisions to be taken, BU-Alfa_Users have generally about two weeks to perform the UAT;

- **Release:** When the UAT has been successfully completed, the new software implementation can be released in Alfa3A, in Alfa4A and in Alfa5. It means that the new functionality or the new dataset is set up in the production environment and that the future data feeding, data collecting, data processing and data reporting will be performed by means of the new software. The production environment is highly delicate and therefore the software releases are always scheduled at the end of the third or fourth week of the month, when BU-Alfa_Users' IT activities are few. In addition, a maintenance service on the new release is provided by any internal and external application systems.

Since the big picture of the planning has been described, the following table (Table 1) shows the list of the milestones planned for Activity1. In order to take into consideration some realistic deadlines, we assume that the BRB was formalized on the 31st of January 2017, the Feasibility Study was successfully concluded on the 31st of March 2017, that means all the feasibility documentations were approved, and at the same date the following plan was validated by all the stakeholders.

***Table 1:** Activity1 planning's example*

ID Task	Milestone	Application System	Task	Unit Owner	Team Owner	Start	End
1	**Design**	Alfa3A Alfa4A	Delivery of the functional and technical documentation	IT-Alfa	**Alfa3_Team Alfa4_Team**	N/A	**21/04**
2	**Design**	Alfa5	Delivery of the technical documentation	IT-Alfa	**Alfa5_Team**	N/A	**21/04**
3	**Design**	BU-Alfa	Validation of the functional documentation	BU-Alfa	**BU-Alfa_Users**	24/04	**28/04**
4	**Design**	IT-Alfa	Validation of the technical documentation	IT-Alfa	**IT-Alfa_Analyst**	24/04	**28/04**
5	**Implementation**	Alfa3A Alfa4A Alfa5	End of software implementation and independent technical tests	IT-Alfa	**Alfa3_Team Alfa4_Team Alfa5_Team**	01/05	**19/05**
6	**SIT**	S.System(III) Alfa3A	System Integration Test – Step1: data feeding to Alfa3A	IT-Alfa	**S.System(III) Alfa3_Team**	N/A	**22/05**
7	**SIT**	Alfa3A	System Integration Test – Step2: Alfa3A data collecting	IT-Alfa	**Alfa3_Team**	22/05	**23/05**
8	**SIT**	Alfa3A Alfa4A	System Integration Test – Step3: data feeding to Alfa4A	IT-Alfa	**Alfa3_Team Alfa4_Team**	23/05	**24/05**
9	**SIT**	Alfa4A	System Integration Test – Step4: Alfa4A data processing	IT-Alfa	**Alfa4_Team**	24/05	**30/05**

10	**SIT**	Alfa4A Alfa5	System Integration Test – Step5: data output to Alfa5	IT-Alfa	**Alfa4_Team Alfa5_Team**	30/05	**31/05**
11	**SIT**	Alfa5	System Integration Test – Step6: Alfa5 data reporting	IT-Alfa	**Alfa5_Team**	31/05	**01/06**
12	**UAT**	BU-Alfa	User Acceptance Test and Business Certification	BU-Alfa	**BU-Alfa_Users**	05/06	**16/06**
13	**Release**	IT-Alfa	Release of the Software in Production environment	IT-Alfa	**Alfa3_Team Alfa4_Team Alfa5_Team**	N/A	**23/06**

3.3.3 BU-Alfa Ventures Execution

Since we take into consideration the assumptions made in paragraph 3.3.2, Activty1 planning (cfr. Table 1) was approved and the execution of the BU-Alfa venture can take place.

From a Project Management point of view, as the Feasibility Study is concluded, the monitoring process is the daily activity that IT-Alfa_PMO and IT-Alfa_Manager must perform. Since the activities on IT-Alfa are many, there is a meeting of two hours every week with all the IT-Alfa stakeholders aimed at drilling down all the projects carried out in IT-Alfa as both operator and contributor. The goals of this weekly meeting are to monitor the activities and to discuss the main issues come up. However, if the problems argued are too complex to address or concern external actors, a dedicated conference will be scheduled. Moreover, if an Alfa#_Team does not meet the deadline shared, IT-Alfa_PMO pushes for delivering the task or for a feedback about possible issues to solve.

Since the positive scenario, where the deadlines are met and the output of Activity1 meets the Business Requirements and the BU-Alfa_Users' expectation as well, has been described in paragraph 3.3.2, we will focus on problem management and the resulting plan execution.

- The **requirement** phase typically does not face relevant issues because BU-Alfa_Users have a clear idea about the Business Requirements since the project is a BU-Alfa venture. A possible hurdle comes up when some specific requirements are not technically feasible or have a very high implementation cost and, as a consequence, or BU-Alfa turns down such requirements or has enough budget to cover the cost and no urgency to develop the project. In the latter case, the time to complete the Feasibility Study widens over the general average of two months.
- Since the Feasibility Study is typically well executed, the **design** phase does not concern many difficulties as well. A negative situation happens when the deadlines are not met by some Alfa#_Team because of a longer documents' drafting or by BU-Alfa_Users due

to some explanation’s requests about the documents’ validation. However, this issue is not critical because the implementation phase actually begins even before the formal process of the functional documents’ validation is completed and therefore this possible delay usually does not impact on the overall activity plan;

- According to the Waterfall lifecycle, each phase is completed and approved before the team moves on the next one, but actually Alfa#_Teams start the **implementation** process as they deliver the design documentations even if, formally, they should wait for BU-Alfa_Users validation. This arrangement allows Alfa#_Teams to spend about one week more for implementing the software aiming at meeting the phase’s deadline. For example, by taking into consideration the Activity1 plan showed in Table 1, Alfa3_Team, Alfa4_Team and Alfa5_Team should initiate the software development the 24^{th} of April and not the 1^{st} of May by gaining one working week. During this phase IT-Alfa_Team has not visibility on the work performed by Alfa#_Teams (Figure 18) and, therefore, each Alfa#_Team has the duty to promptly report any doubt or issue faced. This is a delicate situation because, on one hand, if the problems are not addressed or, even worst, not seen, then the issues will emerge during the test phase (SIT or UAT) and will generate a critical impact on the project completion. On the other hand, the reported problems need for a permanent solution that may be found by consulting the feasibility document and the design documents, or by getting an opinion from BU-Alfa_User. The latter case is always a risk for IT-Alfa because it may generate an extra requirement or even a change requirement from BU-Alfa_Users. However, this critical situation is rare if the Feasibility Study is correctly performed and comprehensively documented.
- The **System Integration Test** (SIT) aims at verifying the goodness of the software development performed during the previous phase by feeding the application systems with the already defined data. The SIT concerns many steps and each tasks’ owner communicates the outcome and the ending of each step to IT-Alfa_Team in order to monitor the process. The data boundary and the modality of data feeding, data processing and data reporting along IT-Alfa were determined by BU-Alfa_Users, and technically confirmed by IT-Alfa_Manager, during the planning procedure. SIT issues may concern different typologies of errors but the schedule generally allows a one or two-days delay (e.g. Table 1 shows a gap of three not-working days between the end of the SIT and the begin of the UAT), however, if the problems concern the processing engine, then the delay can be critical by not only involving the release postponement but also impacting

on the UAT schedule and therefore on the BU-Alfa_Users daily activity. Moreover, a massive delay (about one working week or more) can impact on the other planned System Integration Tests related to other activities, resulting in a double release postponement. Taking into consideration the example of Activity1 (Table 1), the integration test of the Processing Engine Alfa4A is planned from the 24th to the 30th of May and if we assume that Alfa4A is also involved in a SIT for Activity2 starting from the 5th of June, then it is clear that a one-week delay on the SIT of Activity1 will impact on Acticity2 planning as well. In such cases, IT-Alfa_Manager, in the collaboration with IT-Alfa_PMO, push Alfa4_Team for a quick and permanent solution and manage the scheduling of the other activities affected by promptly updating all the stakeholders involved. Moreover, as a contingency solution, the UAT phase can start with a subset of the information and incrementally continue.

- The **User Acceptance Test** (UAT) aims at verifying whether the final output of the new software development meets the Business Requirements of the project and it is performed by BU-Alfa_Users. The UAT modality and its perimeter were determined along with SIT definition during the planning. Since a negative feedback from BU-Alfa_Users can be the worst scenario for the project, IT-Alfa_Manager and IT-Alfa_PMO generally push for a response from Bu-Alfa_Users even before the deadline (e.g. since Activity1 provides UAT from the 5th to the 16th of June, the reminder could be sent on the 12th of May). In case BU-Alfa_Users report an issue, IT-Alfa_Manager and IT-Alfa_Analyst investigate it in order to determine its source and the following scenarios may happen:
    - First, the problem concerns a wrong software interpretation or computation logic, then the application system Alfa# solves it in short term or Alfa# cannot easily rectify the software and the release is postponed until the error is solved;
    - Second, the issue concerns a misinterpretation of the Business Requirements through the fault of BU-Alfa_Users, then BU-Alfa office requires a Change Request (CR) that is a new independent project for which a new Feasibility Study will be developed, and a new effort estimation will be provided as well, resulting in very high reworking cost;
    - Third, the error concerns the data quality depending on the feeding Source System, then the whole SIT is compromised as well as the test output. The Source System takes charge of the errors and schedules the solution that might be in short term (one week) but also in long term (one month). This issue impacts not only

the project planning but also the testing modality because, by postponing the release, data boundary and data cut-off have to change and actually the IT-Alfa application systems have already spent the effort estimated for the first SIT. Therefore, BU-Alfa or pays a new SIT or approves the release and the will perform new UAT that can generate a CR;

- As we have mentioned above, there are some situation where the **release** is postponed, but actually this is not a simple action because the software releases can only be performed at the end of the third or fourth week of the month (cfr. Paragraph 3.3.2). Looking at Table 1, we can see that the final delivery of Activity1 is planned the 23rd of June and, in case of a one-week delay, no postponement is necessary because the planning provides one week of contingency between this deadline and the end of UAT that is scheduled the 16th of June. Moreover, assuming a slightly higher delay (e.g. two weeks), the operative risks are still low because the software release can be performed until the 29th of June (the last day of the month cannot be considered as release date because is not technically feasible due to some IT maintenance services). The critical situation is when the delivery cannot be released within the end of the month and therefore about a three-weeks box is added to the already accumulated delay. Such fault-finding circumstances have risky impacts not only on the project management in IT-Alfa, but also on the economic analysis performed in BU-Alfa.

3.3.4 IT-Alfa Contributions Management

All the main observations and thinking we made in the previous paragraph about the Project Management for a BU-Alfa venture are valid for the IT-Alfa contribution described in paragraph 3.2.2 as well. However, the project contributions have specific features that add value to the thesis' study and are therefore investigated below.

Let's assume that BU-Gamma, because of financial reasons, needs for a new processing functionality that has to be applied on the data coming from Source System (I). Then, since IT-Gamma acquire such data from IT-Alfa, BU-Gamma_Users state their economic needs to BU-Alfa_Users who, consequently, determine whether they are interested at the same business requirements or not. The case in which BU-Alfa_Users are not keen on implementing such new functionality is extremely rare because, on one hand, the BUs have always some economic goals in common and, on the other hand, such situation can cut the project off. Then, we assume that BU-Alfa_Users approves the business requirements stated by BU-Gamma and, as a consequence, they write and formally delivery the BRB for the development of Activity3 to IT-Alfa and to IT-Gamma.

Activity3 has therefore two dataflows in scope, IT-Alfa_Dataflow3 (Figure 26) and IT-Alfa_Dataflow4 (Figure 27), and involves seven application systems (five of IT-Alfa and two of IT-Gamma), two Business Units (BU-Alfa and BU-Gamma) and the Source System (I) (Figure 35). The BRB is written by both the BUs and the budget can be shared as well, or entirely funded by BU-Gamma since the project is a BU-Gamma venture.

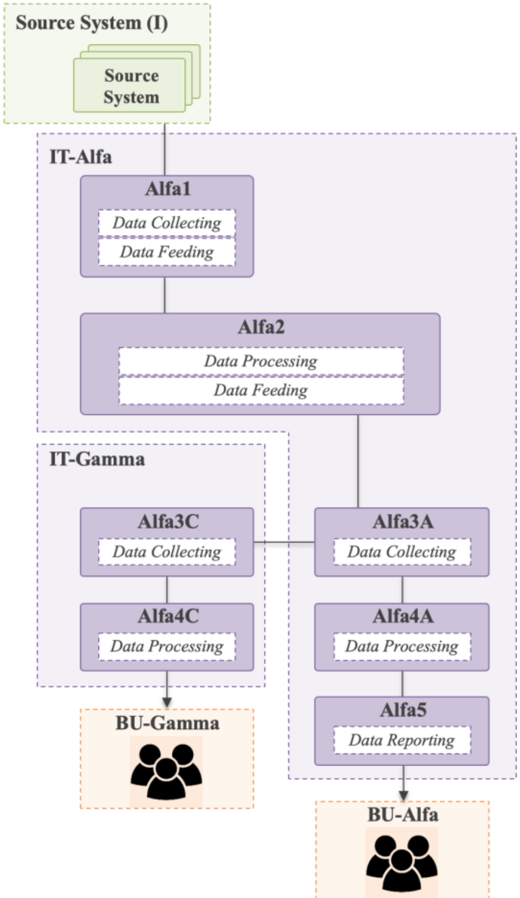

Figure 35: The IT dataflow of Activity3

The Project Management to perform for Activity3 is not easy at all because of the high number of systems and actors involved. Below, we will deep dive into the Waterfall dynamics of Activity3.

- The **requirement** phase of Activity3 is definitely more complex and longer to execute than the one of Activity1 (cfr. Paragraph 3.3.3). The Business Requirements are analyzed in the following steps:

- The first brain storming cycle is focused on the upward part of IT-Alfa architecture (*Alfa1 → Alfa2 → Alfa3A*) and is therefore carried out by IT-Alfa_Team and IT-Gamma_Team with Alfa1_Team, Alfa2_Team and Alfa3A_Team. Any doubt about business decision is reported to both BU-Alfa and BU-Gamma to consequently solve it. The main issue concerns a disagreement between BU-Alfa and BU-Gamma for this first part of data feeding that is in common with IT-Alfa and IT-Gamma. However, these application systems are of IT-Alfa architecture and therefore BU-Gamma_Users should adapt their business requirements to the economic needs of BU-Alfa_Users, even if the project is a BU-Gamma venture and is entirely financed by BU-Gamma_Users. The only alternative for BU-Gamma would be to require a new data feeding from Source System (I) directly toward Alfa3C by cutting IT-Alfa off, but it would be an independent project that would take a lot of money and time and would have an uncertain cost-benefit relationship;
- Second, the data processing requirements in Alfa4A and Alfa4C must be defined, as well as their feeding modality from Alfa3A and Alfa3C and the reporting needs toward the BUs' users (*Alfa3A → Alfa4A → Alfa5 → BU-Alfa* and *Alfa3A → Alfa3C → Alfa 4C → BU-Gamma*). Again, some meetings are scheduled first with the technical teams and then with also the business teams until both the technical and the business requirements are definitively stated. Since Alfa4A and Alfa4C are independent, a disagreement about the data processing required by BU-Alfa_Users with respect to the one required by BU-Gamma_Users is allowed, but strongly discouraged by both IT-Alfa_Manager and IT-Gamma_Manager. The IT teams, indeed, always seek for synergy between the different IT units because technical divergences are often the source of many issues at both technical and organizational levels.
- As the Feasibility Study's document is validated by all the stakeholders, the effort estimation is consolidated and communicated to Alfa_ServiceManager and Gamma_ServiceManger;
- Finally, as the estimated budget is approved, the operative plan of Activity3 is scheduled. Assuming that the BRB was formalized on the 30th of April 2017, that the Feasibility Study was successfully concluded on the 31st of July 2017 and that

August has few working days, Activity3 follows the operative plan shown in Table 2;

Table 2: *Activity3 planning's example*

ID Task	Milestone	Application System	Task	Unit Owner	Team Owner	Start	End
1	**Design Alfa**	Alfa1 Alfa2 Alfa3A Alfa4A	Delivery of the functional and technical documentation	IT-Alfa	**Alfa1_Team** **Alfa2_Team** **Alfa3_Team** **Alfa4_Team**	N/A	**01/09**
2	**Design Gamma**	Alfa3C Alfa4C	Delivery of the functional and technical documentation	IT-Gamma	**Alfa3_Team** **Alfa4_Team**	N/A	**01/09**
3	**Design Alfa**	Alfa5	Delivery of the technical documentation	IT-Alfa	**Alfa5_Team**	N/A	**01/09**
4	**Design Alfa**	BU-Alfa	Validation of the functional documentation	BU-Alfa	**BU-Alfa_Users**	04/09	**08/09**
5	**Design Gamma**	BU-Gamma	Validation of the functional documentation	BU-Gamma	**BU-Gamma_Users**	04/09	**08/09**
6	**Design Alfa**	IT-Alfa	Validation of the technical documentation	IT-Alfa	**IT-Alfa_Analyst**	04/09	**08/09**
7	**Design Gamma**	IT-Gamma	Validation of the technical documentation	IT-Gamma	**IT-Gamma_Analyst**	04/09	**08/09**
8	**Implementation Alfa**	Alfa1 Alfa2 Alfa3A Alfa4A Alfa5	End of software implementation and independent technical tests	IT-Alfa	**Alfa1_Team** **Alfa2_Team** **Alfa3_Team** **Alfa4_Team** **Alfa5_Team**	11/09	**29/09**
9	**Implementation Gamma**	Alfa3C Alfa4C	End of software implementation and independent technical tests	IT-Gamma	**Alfa3_Team** **Alfa4_Team**	11/09	**29/09**
10	**SIT Alfa**	S.System(I) Alfa1	System Integration Test – Step1: data feeding to Alfa1	IT-Alfa	**S.System(I)** **Alfa1_Team**	N/A	**02/10**
11	**SIT Alfa**	Alfa1	System Integration Test – Step2: Alfa1 data collecting	IT-Alfa	**Alfa1_Team**	02/10	**03/10**
12	**SIT Alfa**	Alfa1 Alfa2	System Integration Test – Step3: data feeding to Alfa2	IT-Alfa	**Alfa1_Team** **Alfa2_Team**	03/10	**04/10**
13	**SIT Alfa**	Alfa2	System Integration Test – Step4: Alfa2 data collecting	IT-Alfa	**Alfa2_Team**	04/10	**05/10**
14	**SIT Alfa**	Alfa2 Alfa3A	System Integration Test – Step5: data feeding to Alfa3A	IT-Alfa	**Alfa2_Team** **Alfa3_Team**	05/10	**06/10**
15	**SIT Alfa**	Alfa3A	System Integration Test – Step6: Alfa3A data collecting	IT-Alfa	**Alfa3_Team**	06/10	**09/10**
16	**SIT Alfa**	Alfa3A Alfa4A	System Integration Test – Step7: data feeding to Alfa4A	IT-Alfa	**Alfa3_Team** **Alfa4_Team**	09/10	**10/10**
17	**SIT Alfa**	Alfa4A	System Integration Test – Step8: Alfa4A data processing	IT-Alfa	**Alfa4_Team**	10/10	**13/10**
18	**SIT Alfa**	Alfa4A Alfa5	System Integration Test – Step9: data output to Alfa5	IT-Alfa	**Alfa4_Team** **Alfa5_Team**	13/10	**16/10**
19	**SIT Alfa**	Alfa5	System Integration Test – Step10: Alfa5 data reporting	IT-Alfa	**Alfa5_Team**	16/10	**17/10**

20	**UAT Alfa**	BU-Alfa	User Acceptance Test and Business Certification	BU-Alfa	**BU-Alfa_Users**	17/10	**27/10**
21	**SIT Gamma**	Alfa3A Alfa3C	System Integration Test – Step11: data feeding to Alfa3C	IT-Gamma	**Alfa3_Team**	30/10	**31/10**
22	**SIT Gamma**	Alfa3C	System Integration Test – Step12: Alfa3C data collecting	IT-Gamma	**Alfa3_Team**	31/10	**02/11**
23	**SIT Gamma**	Alfa3C Alfa4C	System Integration Test – Step13: data feeding to Alfa4C	IT-Gamma	**Alfa3_Team** **Alfa4_Team**	02/11	**03/11**
24	**SIT Gamma**	Alfa4C	System Integration Test – Step14: Alfa4C data processing	IT-Gamma	**Alfa4_Team**	03/11	**08/11**
25	**SIT Gamma**	Alfa4C	System Integration Test – Step15: data reporting to BU-Gamma	IT-Gamma	**Alfa4_Team**	08/11	**10/11**
26	**UAT Gamma**	BU-Gamma	User Acceptance Test and Business Certification	BU-Gamma	**BU-Gamma_Users**	13/11	**20/11**
27	**Release**	IT-Alfa IT-Gamma	Release of the Software in Production environment	IT-Alfa IT-Gamma	**Alfa1_Team** **Alfa2_Team** **Alfa3_Team** **Alfa4_Team** **Alfa5_Team**	N/A	**24/11**

- The **design** phase has the same structure described in paragraphs 3.3.2 and 3.3.3. Each Alfa#_Team drafts the functional and/or the technical documents for each application system Alfa#. In addition to what was already explained for IT-Gamma in the example of Activity1, the documents concerning the functional implementation provided in Alfa3C and Alfa4C are reviewed by IT-Gamma_Manager and validated by BU-Gamma_Users whereas the documents about technical and technological aspects are directly validated by IT-Gamma_Analyst. The design process is carried out in parallel by the two IT units;
- IT-Alfa and IT-Gamma perform the **implementation** phase aligned as well. The possible issues to face are the same described in paragraph 3.3.3;
- The **System Integration Test** has an articulate structure divided between the SIT Alfa concerning the test of the IT-Alfa_Dataflow3 and the SIT Gamma concerning the test of the final part of IT-Alfa_Dataflow4 related with IT-Gamma. These two SITs are independent and follow the order described in Table 2: first, SIT Alfa is completed, second, the output of SIT Alfa is tested by BU-Alfa_Users and, only if UAT Alfa is positive, SIT Gamma begins and its output is certified by BU-Gamma_Users. This sequential structure, on one hand, guarantees not to waste the SIT in IT-Gamma in the case there are some errors in the upward part of IT-Alfa software but, on the other hand, widens the time horizon to complete the project. It may indeed happen that, because of lack of time, the SIT Gamma is performed aligned with SIT Alfa; therefore, taking into

consideration Table 2, the task 21 (SIT - step 11) would start in parallel with the task 16 (SIT – step7) the 9th of October with respect to the 30th of October. Consequently, SIT Gamma would end at about the same time of SIT Alfa and the UAT would be performed in parallel by BU-Gamma_Users and BU-Alfa_Users. It is easy to imagine that any issues come up during the SIT generate a critical situation not only for IT-Alfa and IT-Gamma, but also for BU-Alfa and BU-Gamma. The nature and the dynamics of the problems are the same described in paragraph 3.3.3 for the example of the BU-Alfa venture Activity1;

- The planning of the **User Acceptance Tests** depends on the scheduling of their respective SITs. As mentioned above, there are two possible structures: the first one concerns the sequence of SIT Alfa, UAT Alfa, SIT Gamma and UAT Gamma where each phase starts only when the previous is successfully concluded, whereas the second one concerns the parallelism of SIT Alfa and SIT Gamma and as they finish their respective UAT phases begin. The sequential structure is generally preferred because it has a better risk mitigation, whereas the parallel solution is applied when the project has rigid time constrains. The problems that the teams may face are the same described in paragraph 3.3.3 for Activity1, but because Activity3 involves many more stakeholders we will give some examples of problem management:
  - Assuming the tests are planned in a sequential phase as shown in Table 2, a negative UAT Alfa mirrors the same situation showed in paragraph 3.3.3, whereas, in the case UAT Alfa is positive, a negative UAT Gamma should concern an action only by IT-Gamma application systems. However, if BU-Gamma_Users complains about the quality of data feeding, then IT-Alfa investigates the error source and solves it or makes it solve. When the solution is set up, a new SIT is provided but it is performed in parallel between IT-Alfa and IT-Gamma and with the simplest modality (the effort provided for any special test modalities planned runs out with the first SIT). Then, the UAT on the new output is performed by BU-Gamma_Users with the collaboration of BU-Alfa_Users;
  - When the SITs are carried out in parallel, BU-Alfa_Users and BU-Gamma_Users perform the UAT aligned. Any issue reported to the IT units is shared by both the BUs and the problem owner takes it in charge. The next actions are the same described in the bullet point above;

- The **release** activity is performed independently for any application systems in scope. Taking into consideration Activity3, Alfa1_Team releases the new software in Alfa1, Alfa2_Team in Alfa2, Alfa3_Team in Alfa3A and Alfa3C, Alfa4_Team in Alfa4A and Alfa4C, Alfa5_Team in Alfa5. IT-Alfa_Team governs over the software release in the application systems of IT-Alfa and IT-Gamma_Team over IT-Gamma. The dynamics and the impacts concerning the release postponement are the same described in paragraph 3.3.3.

Activity3 represents only one of the multiple IT-Alfa contributions, but its structure and its dynamics are valid for almost all the others. In particular, the Project Management performed for Activity3 is valid also for all the other IT-Alfa contributions to BU-Gamma ventures, that concern different dataflows, as well as the IT-Alfa contributions to BU-Beta, BU-Delta and BU-Epsilon ventures.

However, the IT-Alfa contribution to the Source Systems are slightly different in terms of project scope and requirement phase. On one hand, the contribution projects, which IT-Alfa develops for the Business Units that feeds, are often financed not only by the BU owner of the venture but also by BU-Alfa. On the other hand, when a Source System develops a new product, BU-Alfa never pays for its implementation but anyway writes the Business Requirements for IT-Alfa software development. The economic feasibility is not the only difference because also the planning depends on the Source System owner of the project which states when the new functionality or the new product will be released on its application systems. In such cases, IT-Alfa_Manager determines whether the deadline communicated by the Source System is valid for IT-Alfa as well, or if performing a contingency solution to correctly manage the new software of the Source System until the target solution is carried out.

3.3.5 The Management of Technological Upgrade of IT-Alfa

The technological upgrades of IT-Alfa, that are described in paragraph 3.2.3, concern the hardware activities aimed at improving the machines' capacity and at upgrading the hardware to the newest version.

Assuming that Activity1 and Activity3, described respectively in paragraph 3.3.2/3 and 3.3.4, are both executed, at the end of 2017 the software of IT-Alfa will be upgraded at least twice because of new functionalities and new products. As a consequence, the dataflows will be increased in terms of information number and outputs produced and, aiming at guaranteeing the correct operation of the software, the hardware could need for an extension. Therefore, IT-Alfa_Analyst and the Alfa#_Teams impacted estimate the new required hardware range and communicate it to HW_Team. The execution of such space expansion is quite simple:

- The **requirement** phase states which application system needs for the hardware upgrade, how much is the space range to expand, how much the activity costs and the deadline until which the upgrade must be performed. The planning must be approved by the BU-Alfa_Users in the case the upgrade provides a machines' switching off or any kind of breakdown;
- As the activity is financed, HW_Team performs the hardware upgrade and IT-Alfa_Analyst checks the accuracy of the operation. In case the technical requirements are met, the activity is successfully closed, otherwise a follow-up action is planned, approved and then executed.

However, the Project Management to perform for the hardware version upgrade is not immediate and linear as the situation just described. The switch from the old version of the hardware to the newest means a physical machine change, therefore the old hardware is switched off and replaced by the cutting-edge hardware. Moreover, such kind of activity are scheduled about every four years and are applied to the whole IT system of the Bank. It is therefore easy to imagine the complexity of scheduling the versioning upgrade for every single application systems of any IT units of the Bank.

Since IT-Alfa is a secondary actor in the activity of versioning upgrade because the project is managed and executed by HW_Team, this project typology is not considered in the present work because it will no add value to the thesis objective.

3.3.6 The Control over the Bank Ordinary Reorganization

The project of the Bank ordinary reorganizations, described in paragraph 3.2.4, concerns periodical activities carried out independently by each application system. Assuming Activity4 as the example of a Bank ordinary reorganization, the following phases are provided:

- The Business Requirements are imposed by the Bank and every application system needs for meeting them, with no exception. In the case of IT-Alfa, Activity4 concerns: Alfa1, Alfa2, Alfa3A, Alfa4A, Alfa4. Any doubt about technical requirements is reported to work site owner of Activity4, whereas the problem about business decisions are addressed toward BU-Alfa;

- The design phase requires the documents drafting for reporting the new implementation in each application system. The dynamics and modality of this phase are the same described for Activity1 in paragraph 3.3.2/3;
- IT-Alfa perform the implementation according to the planning provided by the work site and the approach to problem management is the same described in paragraph 3.3.3;
- The test phase is not required for this project typology;
- The release in the production environment must be performed in time with the schedule provided by the work site of Activity4. No delay is admitted because it would generate an impact of regression not only to IT-Alfa, but also to all the other IT Units involved in Activity4.

On one hand, since the time constrains are very rigid, the project management is very important in the Bank ordinary reorganization but, on the other hand, because of the lack of SIT and UAT phases, Activity4 is easy to plan, to monitor and to be executed. Therefore, the thesis considers this kind of project out of scope and will not be further analyzed.

3.4 Waterfall versus Agile Scrum in IT-Alfa

Paragraph 3.3 showed in detail how the four projects' types are managed and only two of them resulted to be in scope for the thesis' analysis. The present paragraph indeed aims at answering the question 'what if we apply Agile Scrum in IT-Alfa' and the analysis will be carried out on two examples: first for the BU-Alfa Venture Activity1 and second for the IT-Alfa Contribution Activity3. For both the activities the two scenarios, Waterfall and Agile, will be investigated by means of the quantitative KPIs described in paragraph 2.4.3.

In order to estimate the Work Cost and the Rework Cost, the following table (Table 3) shows the standard cost rate per man-day of each teams of IT-Alfa, assuming that the internal teams or team members do not impact on the project cost.

Table 3: *List of cost rate per man-day*

Team	Internal/ External	$€/_{ManDay}$
IT-Alfa_Manager	Internal	N/A
IT-Alfa_Analyst	Internal	N/A
IT-Alfa_PMO	External	700
Alfa1_Team	External	300

Alfa2_Team	External	400
Alfa3_Team	External	500
Alfa4_Team	External	900
Alfa5_Team	External	300
IT-Gamma_Team	Internal	N/A
BU-Alfa_Users	Internal	N/A

3.4.1 BU-Alfa Venture in Waterfall

Before introducing the Agile Scrum in a BU-Alfa Venture, we list all the Waterfall phases provided in Activity1, by taking into consideration the planning described in Table 1 and the assumptions made in paragraphs 3.3.2 and 3.3.3, aiming at analyzing all the quantitative KPIs introduced in paragraph 2.4.3 for any operative steps:

- **Requirements** → The 31st of January 2017 BU-Alfa_Users deliver the BRB and we assume it takes the following effort to accomplish the Requirement phase:
  - **Time:** 40 working days. Teams members are part-time. The 31st of March 2017 the Feasibility Study is concluded;
  - **Work Cost:** 24.000€. The effort estimation is 10 man-days for each team involved (Alfa3_Team, Alfa4_Team, Alfa5_Team and IT-Alfa_PMO) resulting in a total of $(500 + 900 + 300 + 700)^{€}/_{ManDay}\ x\ 10ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100-80%. At the very beginning the project can handle any requirements change, but the percentage of adaptability slightly decreases as the feasibility documents are consolidated;
  - **Visibility:** 5 points. All the stakeholders have a full view of the requirements phase's status and its goodness;
  - **Rework Cost:** from 0 to 2.400€ (1/10 of the estimated work cost). In case of reworking, the cost impact is near to zero, because at the project beginning there is enough room for it;
- **Design** → Assuming the best-practice case, the 31st of March 2017 Alfa3_Team, Alfa4_Team and Alfa5_Team begins to design the new software and the following indicators are estimated:

- **Time:** 20 working days. The Design phase not only includes the drafting of technical and functional documents but also their validation. Designers are generally part-time. The 28th of April the Design phase is concluded;
- **Work Cost:** 13.800€. The effort estimation is 8 man-days for Alfa3_Team and Alfa4_Team, 4 man-days for Alfa5_Team and 2 man-days for IT-Alfa_PMO resulting in a total of $(500 + 900)\ ^{€}/_{ManDay}\ x\ 8ManDay + 300^{€}/_{ManDay}\ x\ 4ManDay + 700^{€}/_{ManDay}\ x\ 2ManDay$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 80-50%. The project can handle some requirements change, but the percentage of adaptability definitely decreases as the design documents are validated;
- **Visibility:** 4-2 points. The visibility of the stakeholders strongly decreases during the Design phase from 4 points to less than 2;
- **Rework Cost:** from 2.400€ to 8.500€. In case of reworking, the cost impact increases and can be estimated between 1/5 and 2/5 of the total estimated effort;

- **Implementation** → Assuming the best-practice case, the 24th of April 2017 Alfa3_Team, Alfa4_Team and Alfa5_Team start the new software implementation and the following indicators are estimated:
  - **Time:** 20 working days. The Implementation phase is the longest part of the project in terms of effort and typically starts when the documents' drafting is ended, slightly before the validation deadline. Developers are generally full time. The 19th of May the software implementations are concluded;
  - **Work Cost:** 46.500€. The effort estimation is 30 man-days for Alfa3_Team and Alfa4_Team and 15 man-days for Alfa5_Team resulting in a total of $(500 + 900)\ ^{€}/_{ManDay}\ x\ 30ManDay + 300^{€}/_{ManDay}\ x\ 15ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 40-10%. The project becomes even more rigid and cannot take any basic change in the requirements, however some fine tunings are still allowed;
  - **Visibility:** 1-0 points. The visibility of the activity's status is limited to the single developers for their respective application systems;
  - **Rework Cost:** from 8.500€ to 40.000€. In case of reworking, the cost strongly boosts meanwhile the implementation phase is performed;

- **System Integration Test (SIT)** → Assuming the best-practice case, the 22nd of May 2017 Alfa3_Team, Alfa4_Team and Alfa5_Team start the SIT by performing the sequence of test steps planned and the following indicators are estimated:
  - **Time:** 10 working days. The SIT is well scheduled between the application systems. Tester are generally full-time and supported by an analysts or developers. The 5th of June the output of the SIT must be reported to BU-Alfa_Users;
  - **Work Cost:** 13.100€. The effort estimation is 8 man-days for Alfa3_Team and Alfa4_Team, 4 man-days for Alfa5_Team and 1 man-day for IT-Alfa_PMO resulting in a total of *(500 + 900)* $€/_{ManDay}$ *x 8ManDay + 300*$€/_{ManDay}$ *x 4ManDay + 700€*;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 10-0%. The project is completely rigid and cannot handle any modification. Any further request will be processed as a CR (Change Request);
  - **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams;
  - **Rework Cost:** from 40.000€ to 45.000€. In case of reworking, the cost is already high and slightly increases;
- **User Acceptance Test (UAT)** → Assuming the best-practice case, the 5th of June 2017 BU-Alfa_Users start the UAT and the following indicators are estimated:
  - **Time:** 10 working days. The IT teams' effort is covered in case of necessity. The 16th of June BU-Alfa_Users must validate the software implementation;
  - **Work Cost:** 6.900€. The effort estimation is 4 man-days for Alfa3_Team and Alfa4_Team, 2 man-days for Alfa5_Team and 1 man-day for IT-Alfa_PMO resulting in a total of *(500 + 900)* $€/_{ManDay}$ *x 4ManDay + 300*$€/_{ManDay}$ *x 2ManDay + 700€*;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 0%. The project is completely rigid and cannot handle any modification. Any further request will be processed as a CR (Change Request);
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** up to 50.000€. In case of reworking, the cost is very high;

- **Release** → Assuming the best-practice case, after the 16th of June the IT teams can release the new software and the following indicator is estimated:
    - **ROI:** 100%. Since the UAT was positive, the new software release fully meets the BR.

Looking at the big picture, when Waterfall Project Management does not face any issue, Activity1 totally costs 104.300€ and takes 5 months to be successfully completed. In the average worst-case scenario, Activity1 can cost up to more than 150.000€ and takes 6 months to be fulfilled.

3.4.2 BU-Alfa Venture in Agile Scrum

Taking into consideration the theoretical description of Agile Project Management wrote in paragraph 2.3, we will now assume the Activity1 execution by means of the Scrum methodology aiming at evaluating the resulting KPIs.

Aiming at mirroring the Agile Scrum framework, described in the paragraphs from 2.3.3 to 2.3.6, we assume that Alfa3_Team, Alfa4_Team, Alfa5_Team and IT-Alfa_Analyst represent the Development Team, IT-Alfa_PMO is the Scrum Master and IT-Alfa_Manager is the Product Owner (cfr. Paragraph 2.3.4). These three Scrum Roles are hired full-time for the whole project and the cost rate of the whole Scrum Team is assumed to be about the average of the cost considered in the Waterfall case: $500^{€}/_{ManDay}$. We assume the Development Team has 5 members.

The project is divided into sprints. Since each sprint has a general duration of 2-4 weeks, we assume each sprint takes 1 months to be implemented because IT-Alfa can release new software in production environment only at the end of the third or fourth week of the month. The number of sprints depends on the complexity of meeting the Business Requirements of Activity1 and on the Scrum Team performance as well. We assume the best-practice for Activity1 takes 3 months to complete the project. Moreover, we assume that 1 month mirrors an effort of 20 man-days.

The quantitative KPIs are investigated for the single sprints:

- **Sprint 1** → The 31st of January 2017 BU-Alfa_Users deliver the BRB and the first sprint begins. This phase takes a lot of effort in the building of the Product Backlog. The following indicators are estimated:
    - **Time:** 20 man-days. The first software release is planned on the 24th of February;

- **Work Cost:** 70.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500€/_{ManDay}$ *x 20ManDay x 7 Men;*
- **ROI:** up to 30%. The first release has not significant impact on the software;
- **Adaptability:** 100%. The project can handle any requirements change;
- **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
- **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;

- **Sprint 2** → The 27th of February 2017 the second sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The second software release is planned on the 30th of March;
  - **Work Cost:** 70.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500€/_{ManDay}$ *x 20ManDay x 7 Men;*
  - **ROI:** from 30% to 70%. The second release has significant impact on the software;
  - **Adaptability:** 100%. The project can handle any requirements change;
  - **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
  - **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;
- **Sprint 3** → The 31st of March 2017 the third sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The third software release is planned on the 28th of April;
  - **Work Cost:** 70.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500€/_{ManDay}$ *x 20ManDay x 7 Men;*

- **ROI:** from 70% to 100%. The third release can complete the software development;
- **Adaptability:** 100%. The project can handle any requirements change;
- **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
- **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;

Assuming that the Business Requirements are met by implementing 3 sprints, the Agile Project Management of Activity1 will then cost 210.000€ and takes 3 months. Since Agile Scrum provides an effective governance of the activities, as demonstrated by the KPIs of Adaptability and Visibility, the project results to be successfully completed after the number of sprints planned, three in this case. But if the BU-Alfa_Users requires new Product Backlog Items at the end of Sprint 3, then a fourth sprint would be performed and the total cost of Activity1 rises up to 280.000€ for a 4 months project.

3.4.3 IT-Alfa Contribution in Waterfall

After having analyzed the BU-Alfa venture's performances by applying the Waterfall Project Management and then the Agile Scrum, we will now perform the same exercise for the IT-Alfa contribution by focusing on the example of Activity3 described in the paragraph 3.3.4.

By taking into consideration the Acivtity3 planning showed in Table 2 and the cost rate reported in Table 3, we analyze all the quantitative KPIs introduced in paragraph 2.4.3 for any operative steps:

- **Requirements** → The 30th of April 2017 BU-Gamma_Users and BU-Alfa_Users deliver the BRB to IT-Gamma and IT-Alfa and we assume it takes the following effort to accomplish the Requirement phase:
  - **Time:** 60 working days. Teams' members are part-time. The 31st of July 2017 the Feasibility Study is concluded;
  - **Work Cost:** 52.000€. The effort estimation is 10 man-days for Alfa1_Team, Alfa2_Team and Alfa5_Team, whereas for Alfa3_Team and Alfa4_Team and

IT-Alfa_PMO is 20 man-days, resulting in a total of $(300 + 400 + 300) {€}/_{ManDay}\ x\ 10ManDay + (500 + 900 + 700) {€}/_{ManDay}\ x\ 20ManDay;$

- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100-80%. At the very beginning the project can handle any requirements change, but the percentage of adaptability slightly decreases as the feasibility documents are consolidated;
- **Visibility:** 5 points. All the stakeholders have a full view of the requirements phase's status and its goodness;
- **Rework Cost:** from 0 to 5.200€ (1/10 of the estimated work cost). In case of reworking, the cost impact is near to zero, because at the project beginning there is enough room for it;

- **Design** → Assuming the best-practice case, the 31st of June 2017 Alfa1_Team, Alfa2_Team, Alfa3_Team, Alfa4_Team and Alfa5_Team begins to design the new software and the following indicators are estimated:
  - **Time:** 25 working days. The Design phase not only includes the drafting of technical and functional documents but also their validation. Designers are generally part-time. The 8th of September the Design phase is concluded;
  - **Work Cost:** 27.100€. The effort estimation is 6 man-days for Alfa1_Team and Alfa2_Team, 12 man-days for Alfa3_Team and Alfa4_Team, 4 man-days for Alfa5_Team and 2 man-days for IT-Alfa_PMO resulting in a total of $(300 + 400)\ {€}/_{ManDay}\ x\ 6ManDay + (500 + 900)\ {€}/_{ManDay}\ x\ 14ManDay + 300 {€}/_{ManDay}\ x\ 4ManDay + 700 {€}/_{ManDay}\ x\ 3ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 80-50%. The project can handle some requirements change, but the percentage of adaptability definitely decreases as the design documents are validated;
  - **Visibility:** 4-2 points. The visibility of the stakeholders strongly decreases during the Design phase from 4 points to less than 2;
  - **Rework Cost:** from 5.200€ to 25.000€. In case of reworking, the cost impact increases and can be estimated between 1/5 and 2/5 of the total estimated effort;

- **Implementation** → Assuming the best-practice case, the 4th of September 2017 Alfa1_Team, Alfa2_Team, Alfa3_Team, Alfa4_Team and Alfa5_Team start the new software implementation and the following indicators are estimated:
  - **Time:** 20 working days. The Implementation phase is the longest part of the project in terms of effort and typically starts when the documents' drafting is ended, slightly before the validation deadline. Developers are generally full time. The 29th of September the software implementations are concluded;
  - **Work Cost:** 81.500€. The effort estimation is 30 man-days for Alfa1_Team, Alfa2_Team, 40 man-days for Alfa3_Team and Alfa4_Team and 15 man-days for Alfa5_Team resulting in a total of $(300 + 400)\ {€}/{ManDay}\ x\ 30ManDay$ + $(500 + 900)\ {€}/{ManDay}\ x\ 40ManDay + 300{€}/{ManDay}\ x\ 15ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 40-10%. The project becomes even more rigid and cannot take any basic change in the requirements, however some fine tunings are still allowed;
  - **Visibility:** 1-0 points. The visibility of the activity's status is limited to the single developers for their respective application systems;
  - **Rework Cost:** from 25.000€ to 70.000€. In case of reworking, the cost strongly boosts meanwhile the implementation phase is performed;
- **System Integration Test (SIT)** → Assuming the best-practice case, the 2nd of October 2017 Alfa1_Team, Alfa2_Team, Alfa3_Team, Alfa4_Team and Alfa5_Team start the SIT Alfa and the 30th of October 2017 Alfa3_Team and Alfa4_Team start the SIT Gamma. The test activities follow the operative sequence of test steps planned and the following indicators are estimated:
  - **Time:** a little more than 2 working weeks (10 working days) for SIT Alfa and 2 working weeks (10 working days) for SIT Gamma. Both the SITs are well scheduled between the application systems. Tester are generally full-time and supported by an analysts or developers. The 7th of October the output of the SIT Alfa must be reported to BU-Alfa_Users and the 10th of November the output of the SIT Gamma must be reported to BU-Gamma_Users;
  - **Work Cost:** 29.200€. The effort estimation is 6 man-days for Alfa1_Team and Alfa2_Team, 16 man-days for Alfa3_Team and Alfa4_Team, 4 man-days for Alfa5_Team and 2 man-day for IT-Alfa_PMO resulting in a total of *(300 +*

$400)\ {}^{€}/_{ManDay}\ x\ 6ManDay\ +\ (500\ +\ 900)\ {}^{€}/_{ManDay}\ x\ 16ManDay\ +\ 300{}^{€}/_{ManDay}\ x\ 4ManDay + 700{}^{€}/_{ManDay}\ x\ 2ManDay$;

  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 10-0%. The project is completely rigid and cannot handle any modification. Any further request will be processed as a CR (Change Request);
  - **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams;
  - **Rework Cost:** from 70.000€ to 85.000€. In case of reworking, the cost is already high and increases;
- **User Acceptance Test (UAT)** → Assuming the best-practice case, the 17th of October 2017 BU-Alfa_Users start the UAT Alfa whereas the 13th of November 2017 BU-Gamma_Users start the UAT Gamma, and the following indicators are estimated:
  - **Time:** 3 working weeks overall (8 + 7 working days). The IT teams' effort is covered in case of necessity. On the 27th of October BU-Alfa_Users must validate the software implementation in IT-Alfa and then on the 20th of November BU-Gamma_Users must validate the software implementation in IT-Gamma;
  - **Work Cost:** 14.600€. The effort estimation is 8 man-days for Alfa3_Team and Alfa4_Team, 2 man-days for Alfa1_Team, Alfa2_Team, Alfa5_Team and IT-Alfa_PMO resulting in a total of $(500 + 900)\ {}^{€}/_{ManDay}\ x\ 8ManDay + (300 + 400 + 300 + 700){}^{€}/_{ManDay}\ x\ 2ManDay;$
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 0%. The project is completely rigid and cannot handle any modification. Any further request will be processed as a CR (Change Request);
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** up to 95.000€. In case of reworking, the cost is very high;
- **Release** → Assuming the best-practice case, after the 21st of November the IT teams can release the new software and the following indicator is estimated:
  - **ROI:** 100%. Since both the UATs were positive, the new software release fully meets the BR.

Looking at the big picture, when Waterfall Project Management does not face any issue, Activity3 totally costs 204.400€ and takes 7 months to be successfully completed. In the average worst-case scenario, Activity3 can cost up to more than 250.000€ and takes 8 months to be fulfilled.

3.4.4 IT-Alfa Contribution in Agile Scrum

Taking into consideration the theoretical description of Agile Project Management wrote in paragraph 2.3, we will now assume the Activity3 execution by means of the Scrum methodology aiming at evaluating the resulting KPIs.

Aiming at mirroring the Agile Scrum framework, described in the paragraphs from 2.3.3 to 2.3.6, we assume that Alfa1_Team, Alfa2_Team, Alfa3_Team, Alfa4_Team, Alfa5_Team and IT-Alfa_Analyst represent the Development Team, IT-Alfa_PMO is the Scrum Master and IT-Alfa_Manager is the Product Owner (cfr. Paragraph 2.3.4). These three Scrum Roles are hired full-time for the whole project and the cost rate of the whole Scrum Team is assumed to be about the average of the cost considered in the Waterfall case: $500 €/_{ManDay}$. We assume the Development Team has 9 members.

The project is divided into sprints. Since each sprint has a general duration of 2-4 weeks, we assume each sprint takes 1 months to be implemented because IT-Alfa, as well as IT-Gamma, can release new software in production environment only at the end of the third or fourth week of the month. The number of sprints depends on the complexity of meeting the Business Requirements of Activity3 and on the Scrum Team performance as well. We assume the best-practice for Activity5 takes 5 months to complete the project. Moreover, we assume that 1 month mirrors an effort of 20 man-days.

The quantitative KPIs are investigated for the single sprints:

- **Sprint 1** → The 30th of April 2017 BU-Alfa_Users and BU-Gamma_Users deliver the BRB and the first sprint begins. This phase takes a lot of effort in the building of the Product Backlog. The following indicators are estimated:
  - **Time:** 20 man-days. The first software release is planned on the 26th of May;
  - **Work Cost:** 110.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500 €/_{ManDay}$ *x 20ManDay x 11 Men;*

- **ROI:** up to 20%. The first release has not significant impact on the software;
- **Adaptability:** 100%. The project can handle any requirements change;
- **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
- **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;

- **Sprint 2** → The 29th of May 2017 the second sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The second software release is planned on the 29th of June;
  - **Work Cost:** 110.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500 €/_{ManDay}$ *x 20ManDay x 11 Men;*
  - **ROI:** from 20% to 40%. The second release has still low impact on the software;
  - **Adaptability:** 100%. The project can handle any requirements change;
  - **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
  - **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;
- **Sprint 3** → The 3rd of July 2017 the third sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The third software release is planned on the 28th of July;
  - **Work Cost:** 110.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500 €/_{ManDay}$ *x 20ManDay x 11 Men;*
  - **ROI:** from 40% to 60%. The third release has significant impact on the software development;
  - **Adaptability:** 100%. The project can handle any requirements change;

- **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
- **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;

- **Sprint 4** → The 31th of July 2017 the fourth sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The fourth software release is planned on the 25th of August;
  - **Work Cost:** 110.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500^{€}/_{ManDay}$ *x 20ManDay x 11 Men;*
  - **ROI:** from 60% to 80%. The fourth release aims at almost completing the software development;
  - **Adaptability:** 100%. The project can handle any requirements change;
  - **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
  - **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;
- **Sprint 5** → The 28th of August 2017 the fifth sprint begins. The following indicators are estimated:
  - **Time:** 20 man-days. The fifth software release is planned on the 29th of September;
  - **Work Cost:** 110.000€. The Scrum Team cost is obtained by computing the number of working days estimated times the cost rate times the number of team members: $500^{€}/_{ManDay}$ *x 20ManDay x 11 Men;*
  - **ROI:** from 80% to 100%. The fifth release can complete the software development;
  - **Adaptability:** 100%. The project can handle any requirements change;

- **Visibility:** 5-4 points. All the stakeholders have a full view of the requirements phase's status and its goodness. There is a slight oscillation downward that represents the implementation phase of the daily scrum;
- **Rework Cost:** 0€. Since the Product Backlog may change over time, and new items can be added, the sprint concerns almost no Rework Cost. The trend of the Rework Cost slightly increases as the sprint is ending;

Assuming that the Business Requirements are met by implementing 5 sprints, the Agile Project Management of Activity5 will then cost 550.000€ and takes 5 months. Since Agile Scrum provides an effective governance of the activities, as demonstrated by the KPIs of Adaptability and Visibility, the project results to be successfully completed after the number of sprints planned, five in this case. But if the BU-Alfa_Users or BU-Gamma_Users require new Product Backlog Items at the end of Sprint 5, then a sixth sprint would be performed and the total cost of Activity3 rises up to 660.000€ for a 6 months project.

3.5 Results of the Comparison between Waterfall and Agile

This section aims at evaluating the results of the analysis developed in paragraph 3.4 concerning the Project Management applied on the two projects' examples of BU-Alfa Venture (Activity1) and IT-Alfa Contribution (Activity3) by means of, first, the Waterfall methodology and then the Agile Scrum approach.

Below, Table 4 shows the quantitative results of the analysis by reporting the best scenario and the worst scenario for each Project typology investigated.

Table 4: *Best and worst scenarios for Time and Cost analysis of Acitivty1 and Activity3*

Project Typology	Best Scenario		Worst Scenario	
	Time	Cost	Time	Cost
BU-Alfa Venture Waterfall	5 months	104.300€	6 months	150.000€
BU-Alfa Venture Agile Scrum	3 months	210.00€	4 months	280.000€
IT-Alfa Contribution Waterfall	7 months	193.200€	8 months	250.000€
IT-Alfa Contribution Agile Scrum	5 months	550.000€	6 months	660.000€

3.5.1 Results Analysis

On one hand, if we take into consideration the economic estimations resulted from the analysis, it is clear that implementing Agile Scrum is definitely more expensive than the Waterfall Project Management because we computed that Activity1 in Waterfall costs half the price than in Agile Scrum in both the scenarios (cfr. Table 4). Moreover, Activity3 showed that the Work Cost for implementing the Agile Project Management can be almost three time higher than the Work Cost of the Waterfall approach. The reasons of these results are due to the full-time effort spent by the Agile teams because the Development Team (composed by 5 members in the example of Activity1 and by 9 workers in Activity3), as well as the Scrum Master and the Product Owner, works full-time for the whole duration of the project. In Waterfall methodology instead, the Alfa#_Teams are not full-time allocated on the single project, but they work on multiple activities and, therefore, their cost is well defined by their confined effort.

However, the expensive Agile effort distribution provides a competitive advantage in terms of lower project duration and, moreover, better time-to-delivery. Because Agile Scrum showed not only a shorter timeline to complete the project, but it also constantly provides potentially shippable software increments. Such monthly software releases aim at generating benefits to the final users of BU-Alfa and, in the case of Activity3, of BU-Gamma as well. This turns into an incremental increase of the Return on Investment indicator as showed in paragraphs 3.4.2 and 3.4.4. On the contrary, as a result of the analysis carried out in paragraphs 3.4.1 and 3.4.3, Waterfall approach showed a higher project duration and a zero flat ROI until the end of the project.

Adaptability and Visibility are key elements for ensuring an effective management of the project. When the activity phase concerns high Adaptability and high Visibility, then the risk mitigation is high as well. Since all the Agile sprints showed 5-4 points for Visibility and 100% of Adaptability, Agile Scrum strongly decreases the probability of facing issues and determines an effective government of Activity1 and Activty3 in their whole evolutions. By looking at the other approach, Waterfall Project Management ensures the same performances only for few phases. On one hand, Adaptability goes from 100-80% in the Requirements phase down to 50% at the end of the Design phase until it reaches 0% during the SIT and UAT. On the other hand, Visibility floats from 5-4 points at the beginning of the project down to 1 point during the implementation phase and up again to 4-5 points at the end of the tests.

To conclude, Agile Scrum has better overall performances than the Waterfall, but such efficiency has a tough cost.

3.5.2 No Winner between Agile and Waterfall

By looking at the big picture, we can deduce that the answer of the question 'which is the best Project Management methodology between Agile and Waterfall?' is not univocal. Because, even if Agile Scrum showed excellent performances, not every project is worth the investment Agile requires. Moreover, the Waterfall weakness are known and if the Project Manager is able to promptly address the issues that may come up, Waterfall methodology provides valuable efficiency as well.

Therefore, this thesis' analysis agrees with the extract of the paper [11] reported in paragraph 2.4.1, "Although agile methodologies triumph over traditional ones in several aspects, there are many difficulties in making them work". Moreover, the paper [12], as we already mentioned in paragraph 2.4.1, concluded the analysis between Agile and Waterfall model by stating that "it depends upon the organization which model to choose".

By focusing on the thesis' framework, on one hand, IT-Alfa, because it represents the Information Technology department of the Bank, must guarantee excellent performances and therefore Agile Project Management can be a strategic tool. On the other hand, though, the Bank has very rigid economic willingness and the budget allocated for IT software development is always accurately monitored and never enough available.

In the case of a BU-Alfa Venture as Activity1, applying the Agile Scrum requires a costly investment but compared with the gain in terms of not only efficiency, but also time-to-delivery, it may be worth the expenditure. By quantifying, BU-Alfa should spend about 100.000€ more aiming at obtaining excellent risk mitigation, three incremental software releases and 2 less months for the Business Requirements fulfillment and project completion. It sounds expensive but definitely valuable and, as a result, BU-Alfa will choose the Agile Project Management application on Activity1.

In the case of an IT-Alfa Contribution as Activity3, applying the Agile Scrum looks not as beneficial as in Activity1 because the investment's growth is strongly remarkable, from about 193.200€ to 550.00€, and does not justify time reduction of 3 months as well as the aforementioned performances. As a result, BU-Gamma and BU-Alfa will choose to maintain the Traditional Waterfall Project Management.

Therefore, we can conclude that the thesis' analysis does not declare an overall winner between the Agile and the Waterfall for IT-Alfa, but it rather provides the KPIs aimed at defining which of these two Project Management methodologies fits better with respect to the project itself.

However, the thesis' project aims at investigating a third option that may generate benefits to IT-Alfa: Hybrid Agile.

3.6 Hybrid Agile: Optimization of the Traditional Framework

The present paragraph aims at defining the combination of the Waterfall Life Cycle within the Agile Project Management by applying the Hybrid Agile in the Traditional framework of IT-Alfa.

Since the output of the thesis' analysis developed in the previous paragraphs showed that both Waterfall and Agile have benefits and drawbacks that do not allow to declare a univocal best Project Management methodology, the Hybrid Agile provides a third option that aims at optimizing the Traditional Waterfall model by means of the Agile Scrum mindset.

A similar approach has already been taken into consideration by [15] that writes: "Both traditional and agile approaches have their advantages and disadvantages, if compared to different project characteristics. Approach selection should be handled with care, considering both project characteristics and characteristics of the organizational environment, and it is possible to combine both approaches for the single project and within single methodology".

3.6.1 BU-Alfa in Hybrid Agile

Looking at the big scheme, the Hybrid Agile concerns the mixture of Agile Scrum and Waterfall Life Cycle, where each Agile sprint develops all the Waterfall phases. The sprint is therefore a one-month time box where Requirements, Design, Implementation, Test and Release are performed in a sequential order, as described in the Waterfall theory of paragraph 2.2. The scope of the sprint is limited to one or at least two application systems, structuring them from the Pre-Processing, toward the Processing Engine, until the Post-Processing. Therefore, as Agile Scrum builds the Product Backlog in order to prioritize the Business Requirements related to the whole system aiming at developing the most relevant BR at first, Hybrid Agile structures the BR priority in dependence of the IT architecture of the system in scope and develops the sprints by means of the Traditional Waterfall model.

By taking into consideration Activity1, IT-Alfa_Dataflow8 (cfr. Figure 31) determines the perimeter of application systems: Alfa3A, Alfa4A, Alfa5. Therefore, the following Hybrid Agile execution is provided:

- **Sprint 1.** First sprint focuses on the data feeding from Source System (III) toward Alfa3A and the data collecting to implement in this application system. Assuming that BU-Alfa_Users deliver the BRB on the 31st of January 2017, the Sprint (assuming the standard of 20 working day) will end on the 28th of February 2017 by providing the following phases:
    - **Requirements & Analysis** → The BRs of Alfa3A are analyzed and consolidated in parallel with the functional and technical documentations. We assume it takes the following effort:
        - **Time:** 8 working days.
        - **Work Cost:** 9.600€. The effort estimation is 10 man-days for Alfa3_Team, 2 man-days for Alfa4_Team and 4 man-days for IT-Alfa_PMO, resulting in a total of $500\,{}^{€}/_{ManDay}\ x\ 10ManDay + 900{}^{€}/_{ManDay}\ x\ 2ManDay + 700{}^{€}/_{ManDay}\ x\ 4ManDay$;
        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 4 points. The stakeholders have a full view of the requirements phase's status and its goodness with the exception of Alfa5_Team;
        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **Implementation** → New software implementation and technical tests in Alfa3A:
        - **Time:** 9 working days.
        - **Work Cost:** 9.000€. The effort estimation is 18 man-days for Alfa3_Team, resulting in a total of $500{}^{€}/_{ManDay}\ x\ 18ManDay$;
        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa3_Team;

        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **System Integration Test (SIT)** → No SIT is provided since Alfa3A is the only application system that has implemented the new software;
    - **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the data acquisition of Alfa3A form Source System III:
        - **Time:** 3 working days.
        - **Work Cost:** 1.800€. The effort estimation is 2 man-days for Alfa3_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $500 ^{€}/_{ManDay}$ $x\ 2ManDay + 700€$;
        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **Release** → Assuming the best-practice case, at the end of February 2017, Alfa3_Team releases the new software development in Alfa3A:
        - **ROI:** 20-40%. The release in Alfa3A can be partially completed with respect to the BR (20%) or totally fulfilled (40%);
- **Sprint 2.** Second sprint focuses on the Alfa4A software development and some fine tuning on Alfa3A. The total duration of Sprint 2 concerns the whole month of March, by providing the following phases:
    - **Requirements & Analysis** → The BRs of Alfa4A have been already presented during the Sprint 1, therefore Alfa4_Team has already worked for this phase before Sprint 2 begins. The documentations of Alfa4A are consolidated and, if necessary, the documentations of Alfa3A are updated. We assume it takes the following effort:
        - **Time:** 5 working days (plus 2 working days for Alfa4_Team spent on February).
        - **Work Cost:** 14.400€. The effort estimation is 4 man-days for Alfa3_Team, 10 man-days for Alfa4_Team, 2 man-days for Alfa5_Team

and 4 man-days for IT-Alfa_PMO, resulting in a total of $500^{€}/_{ManDay}$ $x\ 4ManDay + 900\ ^{€}/_{ManDay}\ x\ 10ManDay + 300\ ^{€}/_{ManDay}\ x$ $2ManDay + 700^{€}/_{ManDay}\ x\ 4ManDay$;

- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 4 points. The stakeholders have a full view of the requirements phase's status and its goodness with the exception of Alfa5_Team;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **Implementation** → New software implementation and technical tests in Alfa4A and fine tuning on Alfa3A:

- **Time:** 9 working days.
- **Work Cost:** 20.700€. The effort estimation is 9 man-days for Alfa3_Team and 18 man-days for Alfa4_Team, resulting in a total of $500^{€}/_{ManDay}\ x\ 9ManDay + 900^{€}/_{ManDay}\ x\ 18ManDay$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa3_Team and Alfa4_Team;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **System Integration Test (SIT)** → SIT concerns the data feeding from Alfa3A toward Alfa4A:

- **Time:** 3 working days.
- **Work Cost:** 4.900€. The effort estimation is 3 man-days for Alfa3_Team and Alfa4_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(500 + 900)^{€}/_{ManDay}\ x\ 3ManDay + 700€$;
- **ROI:** 0%. There is not any software delivery;

- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams involved;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the data acquisition of Alfa4A form Alfa3A:
  - **Time:** 3 working days.
  - **Work Cost:** 3.500€. The effort estimation is 2 man-days for Alfa3_Team and IT-Alfa4_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(500 + 900)^{€}/_{ManDay}\ x\ 2ManDay + 700€$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **Release** → Assuming the best-practice case, at the end of March 2017, Alfa3_Team and Alfa4_Team release the new software developments in Alfa3A and Alfa4A respectively:
  - **ROI:** 60-80%. The release in Alfa3A is completed. The new software in Alfa4A can be partially completed with respect to the BR (20%) or totally fulfilled (40%);

- **Sprint 3.** Third sprint focuses on the Alfa5 software development and some fine tuning on Alfa4A because at the end of Sprint 2 Alfa3A has been completely developed. The total duration of Sprint 3 concerns the whole month of April, by providing the following phases:
  - **Requirements & Analysis** → The BRs of Alfa5 have been already presented during the Sprint 2, therefore Alfa5_Team has already worked for this phase before Sprint 3 begins. The documentations of Alfa5 are consolidated and, if

necessary, the documentations of Alfa4A are updated. We assume it takes the following effort:

- **Time:** 5 working days (plus 2 working days for Alfa5_Team spent on March).
- **Work Cost:** 7.300€. The effort estimation is 2 man-days for Alfa4_Team, 9 man-days for Alfa5_Team and 4 man-days for IT-Alfa_PMO, resulting in a total of $900 €/_{ManDay}$ x 2ManDay +$300 €/_{ManDay}$ x 9ManDay + $700 €/_{ManDay}$ x 4ManDay;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 5 points. The stakeholders have a full view of the requirements phase's status and its goodness;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **Implementation** → New software implementation and technical tests in Alfa5 and fine tuning on Alfa4A:

- **Time:** 6 working days.
- **Work Cost:** 14.400€. The effort estimation is 12 man-days for Alfa4_Team and Alfa5_Team, resulting in a total of $(900 + 300) €/_{ManDay}$ x 12ManDay;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa4_Team and Alfa5_Team;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **System Integration Test (SIT)** → SIT concerns the data feeding from Alfa3A toward Alfa4A until Alfa5:

- **Time:** 5 working days.

- **Work Cost:** 5.700€. The effort estimation is 2 man-days for Alfa3_Team and Alfa5_Team, 3 man-days Alfa4_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(500 + 300)^{€}/_{ManDay}$ $x\ 2ManDay$ + $900^{€}/_{ManDay}$ $x\ 3ManDay$ + $700€$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the output of the process regarding data feeding, data collecting, data processing and data reporting:
  - **Time:** 4 working days.
  - **Work Cost:** 4.100€. The effort estimation is 2 man-days for Alfa3_Team, IT-Alfa4_Team and IT-Alfa5_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(500 + 900 + 300)^{€}/_{ManDay}$ $x\ 2ManDay$ + $700€$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **Release** → Assuming the best-practice case, at the end of April 2017, Alfa4_Team and Alfa5_Team release the new software developments in Alfa4A and Alfa5 respectively:
  - **ROI:** 100%. The release in Alfa4A and Alfa5 are completed as well as in Alfa3A.

As one can deduct from the description above, the beginning sprints take more effort for the Business Requirements definition while the last sprints focus the attention on the System Integration Tests and User Acceptance Tests.

3.6.2 IT-Alfa Contribution in Hybrid Agile

Taking into consideration Activity3, we develop the analysis of the previous paragraph also for the IT-Alfa Contribution. Activity3 concerns IT-Alfa_Dataflow3 and IT-Alfa_Dataflow4 (Figure 35) involving seven application systems: Alfa1, Alfa2, Alfa3A, Alfa3C, Alfa4A, Alfa4C and Alfa5. Therefore, the following Hybrid Agile execution is provided:

- **Sprint 1.** First sprint focuses on the data feeding from Source System (I) toward Alfa1 and the data collecting to implement in this application system. Assuming that BU-Alfa_Users deliver the BRB on the 30th of April 2017, the Sprint (assuming the standard of 20 working day) will end on the 26th of May 2017 by providing the following phases:
  - **Requirements & Analysis** → The BRs of Alfa1 are analyzed and consolidated in parallel with the functional and technical documentations. We assume it takes the following effort:
    - **Time:** 8 working days.
    - **Work Cost:** 6.600€. The effort estimation is 10 man-days for Alfa1_Team, 2 man-days for Alfa2_Team and 4 man-days for IT-Alfa_PMO, resulting in a total of $300\,{}^{€}/_{ManDay}\ x\ 10ManDay\ +\ 400{}^{€}/_{ManDay}\ x\ 2ManDay\ +\ 700{}^{€}/_{ManDay}\ x\ 4ManDay$;
    - **ROI:** 0%. There is not any software delivery;
    - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
    - **Visibility:** 4 points. Part of the stakeholders have a full view of the requirements phase’s status and its goodness;
    - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
  - **Implementation** → New software implementation and technical tests in Alfa1:
    - **Time:** 9 working days.
    - **Work Cost:** 5.400€. The effort estimation is 18 man-days for Alfa3_Team, resulting in a total of $300{}^{€}/_{ManDay}\ x\ 18ManDay$;

        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa1_Team;
        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **System Integration Test (SIT)** → No SIT is provided since Alfa1 is the only application system that has implemented the new software;
    - **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the data acquisition of Alfa1 form Source System (I):
        - **Time:** 3 working days.
        - **Work Cost:** 1.300€. The effort estimation is 2 man-days for Alfa1_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $300^{€}/_{ManDay}$ *x 2ManDay + 700€*;
        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **Release** → Assuming the best-practice case, at the end of May 2017, Alfa3_Team releases the new software development in Alfa1:
        - **ROI:** 10-15%. The release in Alfa1 can be partially completed with respect to the BR (10%) or totally fulfilled (15%);
- **Sprint 2.** Second sprint focuses on the Alfa2 software development and some fine tuning on Alfa1. The total duration of Sprint 2 concerns the whole month of June, by providing the following phases:
    - **Requirements & Analysis** → The BRs of Alfa2 have been already presented during the Sprint 1, therefore Alfa2_Team has already worked on this phase before Sprint 2 begins. The documentations of Alfa2 are consolidated and, if

necessary, the documentations of Alfa1 are updated. We assume it takes the following effort:

- **Time:** 5 working days (plus 2 working days for Alfa2_Team spent on May).
- **Work Cost:** 10.000€. The effort estimation is 4 man-days for Alfa1_Team, 10 man-days for Alfa2_Team, 4 man-days for Alfa3_Team and 4 man-days for IT-Alfa_PMO, resulting in a total of $300 €/_{ManDay}$ $x\ 4ManDay + 400\ €/_{ManDay}\ x\ 10ManDay\ +500\ €/_{ManDay}\ x$ $4ManDay + 700€/_{ManDay}\ x\ 4ManDay$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 4 points. Part of the stakeholders have a full view of the requirements phase's status and its goodness;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **Implementation** → New software implementation and technical tests in Alfa2 and fine tuning on Alfa1:

- **Time:** 9 working days.
- **Work Cost:** 9.900€. The effort estimation is 9 man-days for Alfa1_Team and 18 man-days for Alfa2_Team, resulting in a total of $300€/_{ManDay}$ $x\ 9ManDay + 400€/_{ManDay}\ x\ 18ManDay$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa1_Team and Alfa2_Team;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **System Integration Test (SIT)** → SIT concerns the data feeding from Alfa1 toward Alfa2:

- **Time:** 3 working days.
- **Work Cost:** 2.800€. The effort estimation is 3 man-days for Alfa1_Team and Alfa2_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(300 + 400)^{€}/_{ManDay}$ x $3ManDay + 700€$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams involved;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the data acquisition of Alfa2 form Alfa1:

- **Time:** 3 working days.
- **Work Cost:** 2.100€. The effort estimation is 2 man-days for Alfa1_Team and IT-Alfa2_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(300 + 400)^{€}/_{ManDay}$ x $2ManDay + 700€$;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **Release** → Assuming the best-practice case, at the end of June 2017, Alfa1_Team and Alfa2_Team release the new software developments in Alfa1 and Alfa2 respectively:

- **ROI:** 25-30%. The release in Alfa1 is completed (15%). The new software in Alfa2 can be partially completed with respect to the BR (10%) or totally fulfilled (15%);

- **Sprint 3.** Third sprint focuses on the Alfa3A software development and some fine tuning on Alfa2 because at the end of Sprint 2 Alfa1 has been completely developed. The total duration of Sprint 3 concerns the whole month of July, by providing the following phases:

- **Requirements & Analysis** → The BRs of Alfa3A have been already presented during the Sprint 2, therefore Alfa3_Team has already worked for this phase before Sprint 3 begins. The documentations of Alfa3A are consolidated and, if necessary, the documentations of Alfa2 are updated. We assume it takes the following effort:
  - **Time:** 5 working days (plus 4 working days for Alfa3_Team spent on June).
  - **Work Cost:** 10.600€. The effort estimation is 2 man-days for Alfa2_Team, 14 man-days for Alfa3_Team and 4 man-days for IT-Alfa_PMO, resulting in a total of $400\ {€}/{ManDay}\ x\ 2ManDay$ $+500{€}/{ManDay}\ x\ 14ManDay + 700{€}/{ManDay}\ x\ 4ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 5 points. The stakeholders have a full view of the requirements phase's status and its goodness;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **Implementation** → New software implementation and technical tests in Alfa3A and fine tuning on Alfa2:
  - **Time:** 8 working days.
  - **Work Cost:** 11.200€. The effort estimation is 8 man-days for Alfa2_Team and 16 man-days Alfa3_Team, resulting in a total of $400{€}/{ManDay}\ x\ 8ManDay + 500{€}/{ManDay}\ x\ 16ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 1 point. The visibility of the activity's status is limited to the Alfa2_Team and Alfa3_Team;

- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **System Integration Test (SIT)** → SIT concerns the data feeding from Alfa1 toward Alfa2 until Alfa3A:
  - **Time:** 4 working days.
  - **Work Cost:** 4.100€. The effort estimation is 2 man-days for Alfa1_Team and Alfa2_Team, 4 man-days Alfa3_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(300 + 400)€/_{ManDay}\ x\ 2ManDay + 500€/_{ManDay}\ x\ 4ManDay + 700€$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams involved;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the output of the process regarding the dataflow form Alfa1 toward Alfa2, until Alfa3A:
  - **Time:** 3 working days.
  - **Work Cost:** 3.100€. The effort estimation is 2 man-days for Alfa1_Team, IT-Alfa2_Team and IT-Alfa3_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of $(300 + 400 + 500)€/_{ManDay}\ x\ 2ManDay + 700€$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **Release** → Assuming the best-practice case, at the end of July 2017, Alfa2_Team and Alfa3_Team release the new software developments in Alfa2 and Alfa3A respectively:
    - **ROI:** 40-45%. The releases in Alfa1 and Alfa2 are completed (30%). The new software in Alfa3A can be partially completed with respect to the BR (10%) or totally fulfilled (15%);

- **Sprint 4.** Fourth sprint focuses on two parallel streams: one about the Alfa3C software development plus some fine tuning on Alfa3A concerning the Alfa3C feeding, and the other one about the Alfa4A development plus some fine tuning on Alfa3A concerning the Alfa4A feeding. The total duration of both the streams of Sprint 4 concerns the whole month of August, by providing the following phases:
  - **Requirements & Analysis** → The BRs of Alfa4A and Alfa3C have been already presented during the Sprint 3, therefore Alfa4_Team and Alfa3_Team have already worked for this phase before Sprint 4 begins. The documentations of Alfa4A and Alfa3C are consolidated and, if necessary, the documentations of Alfa3A are updated. We assume it takes the following effort:
    - **Time:** 4 working days (plus 4 working days for Alfa4_Team and Alfa3_Team spent on July).
    - **Work Cost:** 23.000€. The effort estimation is (4+12) man-days for Alfa3_Team, 12 man-days for Alfa4_Team and 6 man-days for IT-Alfa_PMO, resulting in a total of $500\ €/_{ManDay}\ x\ 16ManDay\ +900€/_{ManDay}\ x\ 12ManDay + 700€/_{ManDay}\ x\ 6ManDay$;
    - **ROI:** 0%. There is not any software delivery;
    - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
    - **Visibility:** 5 points. The stakeholders have a full view of the requirements phase's status and its goodness;
    - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
  - **Implementation** → New software implementation and technical tests in Alfa4A and in Alfa3C, and fine tuning on Alfa3A:
    - **Time:** 8 working days.

- **Work Cost:** 26.400€. The effort estimation is (8+16) man-days for Alfa3_Team and 16 man-days Alfa4_Team, resulting in a total of $500 ^{€}/_{ManDay}$ *x 24ManDay + 900* $^{€}/_{ManDay}$ *x 16ManDay*;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1 point. The visibility of the activity's status is limited to Alfa3_Team and Alfa4_Team;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **System Integration Test (SIT)** → SIT concerns the data feeding from Alfa2 toward Alfa3A until Alfa4A and Alfa3C:

- **Time:** 5 working days.
- **Work Cost:** 8.800€. The effort estimation is 2 man-days for Alfa2_Team, (2+4) man-days for Alfa3_Team, 4 man-days for Alfa4_Team and 2 man-days for IT-Alfa_PMO, resulting in a total of 400 $^{€}/_{ManDay}$ *x 2ManDay + 500* $^{€}/_{ManDay}$ *x 6ManDay + 900* $^{€}/_{ManDay}$ *x 4ManDay + 700€ x 2ManDay*;
- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams involved;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

o **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the output of the process regarding the dataflow form Alfa2 toward Alfa3A until Alfa4A. Moreover, the UAT concerns, in parallel, the reporting to Bu-Gamma_Users as well, about the output of the process regarding the dataflow form Alfa2 toward Alfa3A until Alfa3C:

- **Time:** 3 working days.

        - **Work Cost:** 5.500€. The effort estimation is 2 man-days for Alfa2_Team, and IT-Alfa4_Team, 4 man-days for IT-Alfa3_Team and 1 man-day for IT-Alfa_PMO, resulting in a total of *(300 + 900)* $\text{€}/_{ManDay}$ *x 2ManDay + 500* $\text{€}/_{ManDay}$ *x 4ManDay + 700€*;
        - **ROI:** 0%. There is not any software delivery;
        - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
        - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
        - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
    - **Release** → Assuming the best-practice case, at the end of August 2017, Alfa3_Team and Alfa4_Team release the new software developments in Alfa3A/ Alfa3C and Alfa4A respectively:
        - **ROI:** 60-70%. The releases in Alfa1, Alfa2 and Alfa3A are completed (45%). The new software in Alfa3C can be partially completed with respect to the BR (5%) or totally fulfilled (10%). The new software in Alfa4A can be partially completed with respect to the BR (10%) or totally fulfilled (15%);
- **Sprint 5.** Fifth sprint focuses on two parallel streams: one about the Alfa4C software development plus some fine tuning on Alfa3C, and the other one about the Alfa5 development plus some fine tuning on Alfa4A. The total duration of both the streams of Sprint 5 concerns the whole month of September, by providing the following phases:
    - **Requirements & Analysis** → The BRs of Alfa5 and Alfa4C have been already presented during the Sprint 4, therefore Alfa5_Team and Alfa4_Team have already worked on this phase before Sprint 5 begins. The documentations of Alfa5 and Alfa4C are consolidated and, if necessary, the documentations of Alfa3C and Alfa4A are updated. We assume it takes the following effort:
        - **Time:** 4 working days (plus 4 working days for Alfa5_Team and Alfa4_Team spent on August).
        - **Work Cost:** 22.400€. The effort estimation is 4 man-days for Alfa3_Team, (2+12) man-days for Alfa4_Team, 12 man-days for

Alfa5_Team and 6 man-days for IT-Alfa_PMO, resulting in a total of $500\ {}^{€}/_{ManDay}\ x\ 4ManDay + 900\ {}^{€}/_{ManDay}\ x\ 14ManDay + 300{}^{€}/_{ManDay}\ x\ 12ManDay + 700{}^{€}/_{ManDay}\ x\ 6ManDay$;

- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 5 points. The stakeholders have a full view of the requirements phase's status and its goodness;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **Implementation** → New software implementation and technical tests in Alfa5 and in Alfa4C and fine tuning on Alfa4A and Alfa3C:
  - **Time:** 8 working days.
  - **Work Cost:** 30.400€. The effort estimation is 8 man-days for Alfa3_Team, (8+16) man-days for Alfa4_Team and 16 man-days Alfa5_Team, resulting in a total of $500\ {}^{€}/_{ManDay}\ x\ 8ManDay + 900{}^{€}/_{ManDay}\ x\ 24ManDay + 300{}^{€}/_{ManDay}\ x\ 16ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 1 point. The visibility of the activity's status is limited to Alfa3_Team, Alfa4_Team and Alfa5_Team;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **System Integration Test (SIT)** → SIT concerns the (parallel) data feeding from Alfa3A toward Alfa3C until Alfa4C and toward Alfa4A until Alfa5:
  - **Time:** 5 working days.
  - **Work Cost:** 10.300€. The effort estimation is 6 man-days for Alfa3_Team, (2+4) man-days for Alfa4_Team, 4 man-days for Alfa5_Team and 2 man-days for IT-Alfa_PMO, resulting in a total of

$500\ {€}/_{ManDay}\ x\ 6ManDay + 900\ {€}/_{ManDay}\ x\ 6ManDay + 300{€}/_{ManDay}\ x\ 4ManDay + 700€\ x\ 2ManDay$;

- **ROI:** 0%. There is not any software delivery;
- **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
- **Visibility:** 1-4 points. The visibility of the activity's status quickly increases by developing the tests to all the IT teams involved;
- **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;

- **User Acceptance Test (UAT)** → The UAT concerns the reporting to BU-Alfa_Users about the output of Alfa5. Moreover, the UAT concerns, in parallel, the reporting to Bu-Gamma_Users about the output of Alfa4C:
  - **Time:** 3 working days.
  - **Work Cost:** 8.200€. The effort estimation is 4 man-days for Alfa3_Team, Alfa4_Team and IT-Alfa5_Team, and 2 man-days for IT-Alfa_PMO, resulting in a total of $(500 + 900 + 300){€}/_{ManDay}\ x\ 4ManDay + 700{€}/_{ManDay}\ x\ 2ManDay$;
  - **ROI:** 0%. There is not any software delivery;
  - **Adaptability:** 100%. The Adaptability is maximum because of the sprint methodology;
  - **Visibility:** 5 points. The visibility of the activity's status is known by all the stakeholders (IT and Business) as well as the goodness of the project;
  - **Rework Cost:** 0€. The Rework Cost is null because of the sprint methodology;
- **Release** → Assuming the best-practice case, at the end of September 2017, Alfa3_Team, Alfa4_Team and Alfa5_Team release the new software developments in Alfa3C, Alfa4A/ Alfa4C and Alfa5 respectively:
  - **ROI:** 100%. The releases in Alfa5 and Alfa4C are completed as well as in Alfa1, Alfa2, Alfa3A, Alfa3C and Alfa4A.

3.6.2 Hybrid Agile versus Waterfall & Agile Scrum

Looking at the big picture, Hybrid Agile approach manages Activity1 with a total cost of 95.400€ and a time-to-delivery equal to 3 months. Such result is definitely remarkable if compared with the time and cost estimation, reported in table 4, of the previous analysis about Waterfall and Agile Scrum, whereas in the worst scenario we assume that Activity1 duration widens of 1 month and the total cost increases of about the average of the sprints' costs ($^{95.000€}/_{3}$), resulting in an investment of 127.200€ for a 4 months project. However, before praising the Hybrid Agile methodology, we must look at Table 5 that provides an accurate comparison between the BU-Alfa venture development in Waterfall and in Hybrid Agile.

Table 5: *Comparison Waterfall vs Hybrid Agile in BU-Alfa Venture Activity1*

Phases	Waterfall			Hybrid Agile		
	Cost (€)	Effort (m/d)[1]	Time (wd)[2]	Cost (€)	Effort (m/d)	Time (wd)
Requirements	24.000€	10 m/d Alfa3 10 m/d Alfa4 10 m/d Alfa5 10 m/d PMO	40 wd	31.300€	14 m/d Alfa3 14 m/d Alfa4 11 m/d Alfa5 12 m/d PMO	18 wd
Design	13.800€	8 m/d Alfa3 8 m/d Alfa4 4 m/d Alfa5 2 m/d PMO	20 wd			
Implementation	46.500€	30 m/d Alfa3 30 m/d Alfa4 15 m/d Alfa5	20 wd	44.100€	27 m/d Alfa3 30 m/d Alfa4 12 m/d Alfa5	24 wd
SIT	13.100€	8 m/d Alfa3 8 m/d Alfa4 4 m/d Alfa5 1 m/d PMO	10 wd	10.600€	5 m/d Alfa3 6 m/d Alfa4 2 m/d Alfa5 3 m/d PMO	8 wd
UAT	6.900€	4 m/d Alfa3 4 m/d Alfa4 2 m/d Alfa5 1 m/d PMO	10 wd	9.400€	6 m/d Alfa3 4 m/d Alfa4 2 m/d Alfa5 3 m/d PMO	10 wd
TOTAL	**104.300€**	**60 m/d Alfa3 60 m/d Alfa4 35 m/d Alfa5 14 m/d PMO**	**100 wd**	**95.400€**	**52 m/d Alfa3 54 m/d Alfa4 27 m/d Alfa5 18 m/d PMO**	**60 wd**

Table 5 shows that, looking at the big scheme, Hybrid Agile represents the crushing version of Waterfall because the total efforts of the teams are nearly equal, also justifying the similar estimation of the total cost. It means that the Hybrid Agile approach of structuring the project into sprints provides the same time-to-delivery result of crashing the Waterfall model, passing

[1] m/d → man-day
[2] wd → working day

from a 5 months project duration to only 3 months. However, when we simply crash the Waterfall model, the Work Cost strongly increases but this does not happen in Hybrid Agile because of the application of the sprint methodology as Agile Scrum teaches. Moreover, the sprint organization strongly increases the Adaptability during the project execution, since there are more than one possibility of testing and releasing the new software. For example, if after the second release of Sprint 2 Alfa3_Team faces some issues in Alfa3A, then the team still has the time to implement solutions, to test them in system environment and to release them into production environment. Moreover, BU-Alfa_Users have an excellent visibility at the end of each Sprint that allows them to better write, or even modify and integrate, the Business Requirements. Such efficacy mirrors not only excellent Project Management performances, but also a decrease in work-days estimation, because the software developments are clearer and clever and there is an outstanding risk mitigation.

This conclusion mirrors the results of the Hybrid Agile application on Activity3 as well. The total cost of the 5 months project is equal to 202.100€ in the best-practice scenario, whereas, if we take into consideration an additional sprint and release, we may estimate a cost improvement of about the average of the sprints' costs ($^{202.100€}/_{5}$), resulting in an investment of 242.500€ for a 6 months project.

Table 6: *Comparison Waterfall vs Hybrid Agile in IT-Alfa Contribution Activity3*

<table>
<tr><th rowspan="2">Phases</th><th colspan="3">Waterfall</th><th colspan="3">Hybrid Agile</th></tr>
<tr><th>Cost (€)</th><th>Effort (m/d)[3]</th><th>Time (wd) [4]</th><th>Cost (€)</th><th>Effort (m/d)</th><th>Time (wd)</th></tr>
<tr><td>Requirements</td><td>52.000€</td><td>10 m/d Alfa1
10 m/d Alfa2
20 m/d Alfa3
20 m/d Alfa4
10 m/d Alfa5
20 m/d PMO</td><td>60 wd</td><td rowspan="2">72.600€</td><td rowspan="2">14 m/d Alfa1
14 m/d Alfa2
38 m/d Alfa3
26 m/d Alfa4
12 m/d Alfa5
24 m/d PMO</td><td rowspan="2">26 wd</td></tr>
<tr><td>Design</td><td>27.100€</td><td>6 m/d Alfa1
6 m/d Alfa2
14 m/d Alfa3
14 m/d Alfa4
4 m/d Alfa5
3 m/d PMO</td><td>25 wd</td></tr>
<tr><td>Implementation</td><td>81.500€</td><td>30 m/d Alfa1
30 m/d Alfa2
40 m/d Alfa3
40 m/d Alfa4
15 m/d Alfa5</td><td>20 wd</td><td>83.300€</td><td>27 m/d Alfa1
26 m/d Alfa2
48 m/d Alfa3
40 m/d Alfa4
16 m/d Alfa5</td><td>42 wd</td></tr>
<tr><td>SIT</td><td>29.200€</td><td>6 m/d Alfa1
6 m/d Alfa2
16 m/d Alfa3</td><td>10 wd
10 wd</td><td>26.000€</td><td>5 m/d Alfa1
7 m/d Alfa2
16 m/d Alfa3</td><td>17 wd</td></tr>
</table>

[3] m/d → man-day
[4] wd → working day

		16 m/d Alfa4 4 m/d Alfa5 2 m/d PMO			10 m/d Alfa4 4 m/d Alfa5 6 m/d PMO	
UAT	14.600€	2 m/d Alfa1 2 m/d Alfa2 8 m/d Alfa3 8 m/d Alfa4 2 m/d Alfa5 2 m/d PMO	8 wd 7 wd	20.200€	6 m/d Alfa1 6 m/d Alfa2 10 m/d Alfa3 6 m/d Alfa4 4 m/d Alfa5 6 m/d PMO	15 wd
TOTAL	**204.400€**	**54 m/d Alfa1 54 m/d Alfa2 98 m/d Alfa3 98 m/d Alfa4 35 m/d Alfa5 27 m/d PMO**	**140 wd**	**202.100€**	**52 m/d Alfa1 53 m/d Alfa2 112m/d Alfa3 82 m/d Alfa4 36 m/d Alfa5 36 m/d PMO**	**100 wd**

Anyway, Hybrid Agile cannot compete against Agile Scrum in terms of overall performances because Scrum involves actors hired full-time and the remarkable output of Agile Project Management has been proved among many projects. However, this analysis shows that Hybrid Agile takes part of the Scrum advantages with a very competitive final cost, resulting in an effective optimization of the Traditional framework of IT-Alfa. Moreover, for the execution of both BU-Alfa Venture and IT-Alfa Contribution, Hybrid Agile results to be the best option over both the Waterfall and the Agile Project Management.

By taking into consideration the applicability, one can see that IT-Alfa perfectly fits the Hybrid Agile approach, whereas Agile Scrum may be complex to adopt. On one hand, pure Agile methodologies represents a mindset change as well as a revolution of the activities management (e.g. new roles, new documents), facing issues and complexity when the project is executed and facing difficulties to master the Agile efficiency. On the other hand, Hybrid Agile concerns the already know Waterfall phases with a different process flow, resulting easier and clever to be applied in IT-Alfa.

The limit of Hybrid Agile is that is based on IT-Alfa architecture and it may show negative performances when applied in other IT units that have different structures. So far, we accept this limit by assuming such analysis as future development.

The figures below provide a graphic representation of the KPIs' trend in Hybrid Agile with respect to Waterfall and Agile Scrum.

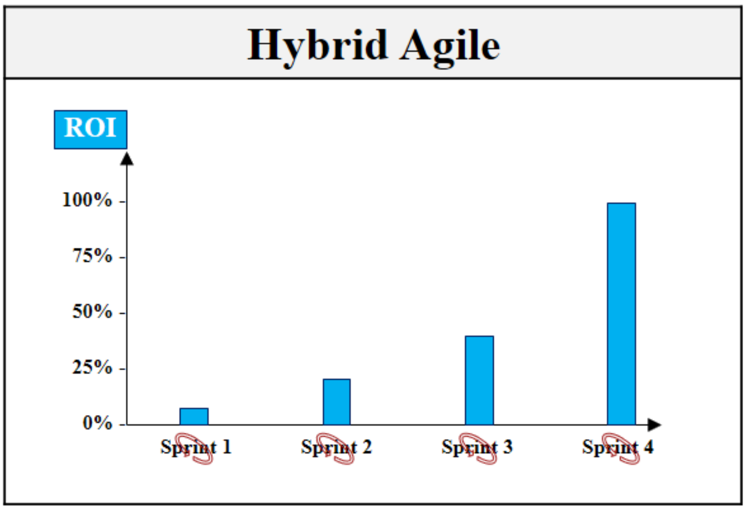

Figure 36: *Hybrid Agile - Return On Interest*

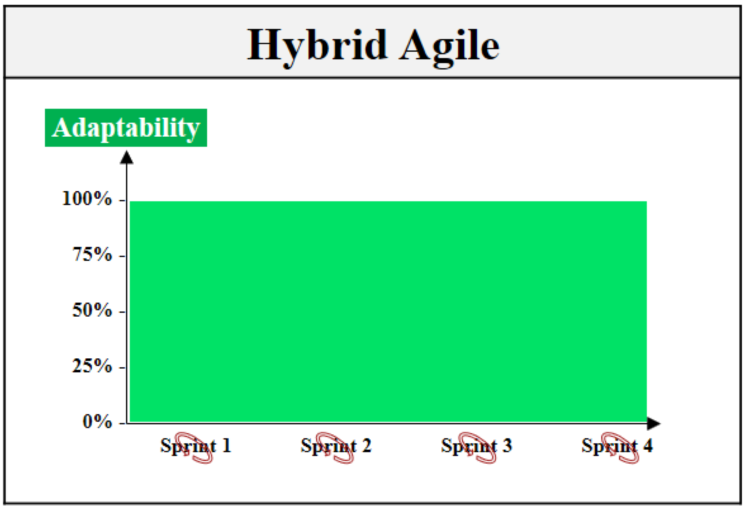

Figure 37: *Hybrid Agile – Adaptability*

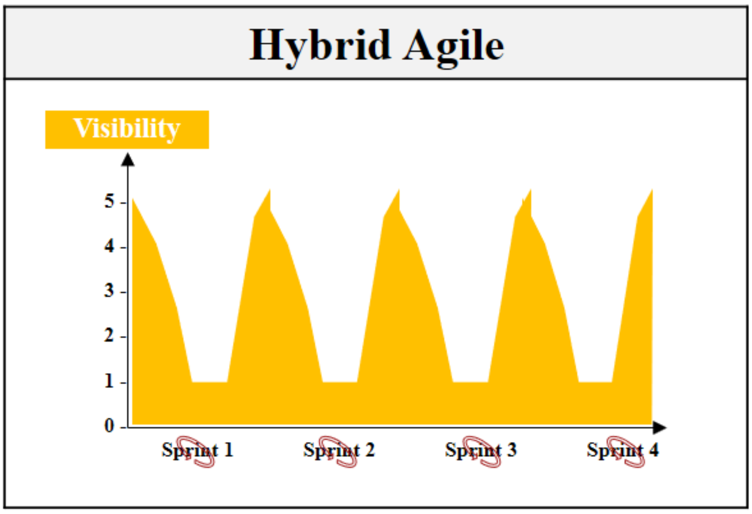

Figure 38: *Hybrid Agile – Visibility*

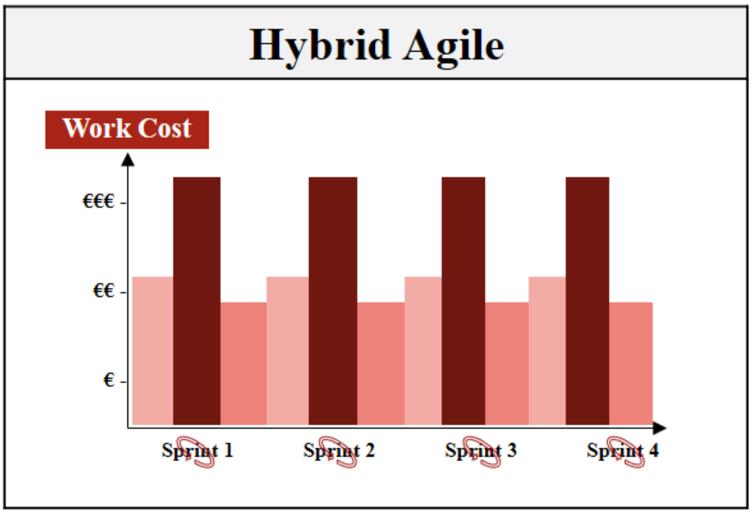

Figure 39: *Hybrid Agile - Work Cost*

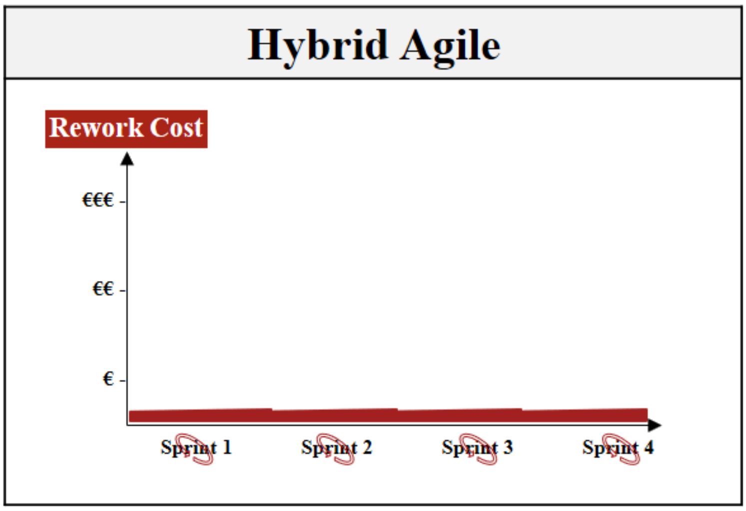

Figure 40: *Hybrid Agile - Rework Cost*

4. Conclusion

The last chapter aims at travelling through the thesis work again to show the conceptual plan that led to the following conclusions.

First of all, the present work provided the theoretical fundaments about the Waterfall Life Cycle and the Agile Project Management. By comparing these underlying principles, the thesis identified five Key Performance Indicators aimed at quantifying the differences in term of effectiveness between the two aforementioned Project Management methodologies: Return on Interest, Adaptability, Visibility, Work Cost and Rework Cost, plus the implied parameter of Time.

Secondly, the thesis described the IT Banking System framework, explaining and showing the architecture of IT-Alfa, the analyzed IT bank department of the case study. Then, the rebus sic stantibus Project Management dynamics were explained concerning four different typologies of activities, of which only two were accepted to be the subjects of the study: BU-Alfa Venture and IT-Alfa Contribution.

The thesis, then, takes into consideration the examples of Activity1 as BU-Alfa Venture and of Acitivty3 as IT-Alfa Contribution for investigating the KPIs' trends by applying on both the cases, first, the Waterfall model and, next, the Agile Scrum. This analysis aimed at investigating whether the Agile Scrum results to be more suitable and advantageous with respect to the Waterfall model. The results showed that Agile Scrum is favorable for Activity1 but is not proper for Activity2 that prefers the Traditional Waterfall approach, because, even if Agile Scrum showed excellent performances, not every project is worth the investment Agile requires. As a consequence, the thesis' study demonstrated in a quantitative way that there is not an univocal best Project Management methodology, but rather that depends upon the organization and the nature of the project which model to choose.

In addition to this first important conclusion, the thesis' project aimed at finding a third solution of Project Management tailored for IT-Alfa: the Hybrid Agile. The approach of Hybrid Agile concerns the mixture of Agile Scrum and Waterfall Life Cycle, where each Agile sprint develops all the Waterfall phases by focusing on the software developments of at least two application systems. The same analysis, before conducted for Activity1 and Activity3 in the Waterfall and in the Agile scenarios, was carried out by applying the Hybrid Agile by means of the aforementioned KPIs. As a result, Hybrid Agile showed remarkable performances in term of both costs and benefits, due to its Waterfall-like effort distribution and because of the advantages provided by the Sprint methodologies. Therefore, since the thesis' analysis showed that, for the execution of both BU-Alfa Venture and IT-Alfa Contribution, Hybrid Agile results to be the best option over both the Waterfall and the Agile Project Management, the thesis concludes that the hybrid application of the Agile Project Management in the IT banking system can optimize the traditional framework.

Furthermore, the case study of the Information Technology department of the Bank highlighted the importance and the needs for the application of an effective Project Management methodology that provides cost reduction, risk mitigation and organization benefits.

Finally, we suggest, as future development of the study, to implement the Hybrid Agile in a real project in IT-Alfa aiming at investigating the issues faced as well as the results achieved.

List of Figures

List of Tables

References

[1] M. Mukherjee and S. Roy, “Feasibility Studies and Important Aspect of Project Management”, *International Journal of Advanced Engineering and Management*, Vol. 2, No. 4, pp. 98-100, 2017

[2] L. Hales, “Agile vs. Waterfall Project Management”, *Quantitative Analysis: Tutoring Solution*, Chapter 2, lesson 14

[3] D. A. Hughey, “Comparing Traditional Systems Analysis and Design with Agile Methodologies”, University of Missouri-St. Louis, 2009

[4] “Benefici del modello a cascata nel project management”, Project Management Center. http://www.humanwareonline.com [accessed the 14/01/2019]

[5] A. Olic, “Waterfall Project Management Methodology”, *Project Management*, 2017

[6] K. Beck, A. Cockburn, R. Jeffries, and J. Highsmith, “The Agile Manifesto”, Agile manifesto web site, 2001. http://www.agilemanifesto.org [accessed the 14/01/2019]

[7] Sixth annual state of agile survey: State of agile development. Atlanta, GA: VersionOne

[8] K. S. Rubin, “A Visually Rich Overview of the Scrum Framework”, Scrum Alliance, 2016

[9] S. Spalek, “Traditional vs. Modern Project Management Methods. Theory and Practice”, *Proceeding of 21st International Scientific Conference Economics and Management*, May 19-20, Brno, Czech Republic, 2016

[10] D. & J. Fernandez, “Agile project management - agilism versus traditional approaches”, *Journal of Computer Information Systems*, Vol. 49, No. 2, PP. 10-17, 2008

[11] M. Stoica, M. Mirces, and B. Ghilic-Micu, “Software Development: Agile vs Traditional”, Informatica Economica, Vol. 12, No. 4, 2013

[12] S. Balaji and M. Sundararajan Murugaiyan, “Waterfall Vs V-Model vs Agile: A comparative study on SDLC”, International Journal of Information Technology and Business Management, Vol. 2, No. 1, 2012

[13] J. Highsmith and A. Cockburn, “Agile Software Development: The Business of Innovation”, *IEEE Computer*, Vol. 34, No. 9, pp. 120-127, 2001

[14] M. A. Awad, “A comparison between Agile and Traditional Software Development Methodologies”, University of Western Australia, 2005

[15] M. Špundak, “Mixed agile/traditional project management methodology – reality or illusion?”, *Procedia of Social and Behavioral Sciences*, Vol. 119, pp. 939 – 948, 2014

[16] https://www.tutorialspoint.com/sdlc/index.htm [accessed the 14/01/2019]

[17] PMBOK Guide, “Project Management Body of Knowledge”, 4th Edition, Newtown Square, 2008

[18] K. Schwaber and J. Sutherland, “The Definitive Guide to Scrum: The Rules of the Game”, 2017

[19] J. Highsmith and A. Cockburn, “Agile Software Development: The People Factor”, *IEEE Computer*, Vol. 34, No. 11, pp. 131-133, 2001

[20] M. Doyle, L. Williams, M. Cohn, and K. S. Rubin, “Agile Software Development in Practice”, *Agile Processes in Software Engineering and Extreme Programming*, pp. 32-45, 2014