Module 4. Agile Methodologies

Unit 4.1 Introduction to Agile Methodologies

Introduction

The concept of *agile methodologies* is applied more and more every day in companies and startups, since they allow adapting the working style to the conditions of the project, thus obtaining greater flexibility and speed of response to adapt the project and its development to the needs of the client and the specific circumstances of the environment. It could be said, then, that ventures that build on these methodologies manage their projects in an effective way, reducing costs and increasing their productivity.

This methodology arises in the field of software development, but it is applicable to any industry and sector.

Let's quickly look at some of the basic principles of agile methodologies presented in the article “Las 8 grandes ventajas de las metodologías ágiles” [The 8 Great Advantages of Agile Methodologies] (Martínez, 2014).

1) Agile methodologies improve customer satisfaction, since the customer will be involved and committed throughout the project. The client will be informed of the progress of the development at each stage. In this way, the customer can add their experience, comments or requests for adjustments to optimize the characteristics of the final product. Therefore, numerous misunderstandings can be avoided as the customer will have, at all times, a complete overview of the state of the product.

2) Likewise, they improve the motivation and involvement of the work team. But this improvement is not accidental: agile methodologies allow all team members to know the status of the project at any time. Commitments are negotiated and accepted by all team members and the ideas of any team member are taken into account.

3) Agile processes allow saving both time and costs. Agile development works more efficiently and quickly than other methodologies. Moreover, these processes focus on strictly complying with the budget and deadlines agreed when defining and planning the project.

4) Agile methodologies allow working with greater speed and efficiency, making partial but functional deliveries of the product. In this way, it is

possible to deliver a functional version of the product in the shortest possible time.

5) Thanks to the partial deliveries (focused on delivering first those functionalities that really add value) and the involvement of the customer, it will be possible to eliminate those unnecessary characteristics of the product.

6) Agile methodologies allow improving the quality of the product. The continuous interaction between the members of the entrepreneurial team and the customers aims to ensure that the final product is exactly what the customer wants and needs. In addition, this approach allows embracing excellence, resulting in a superior product.

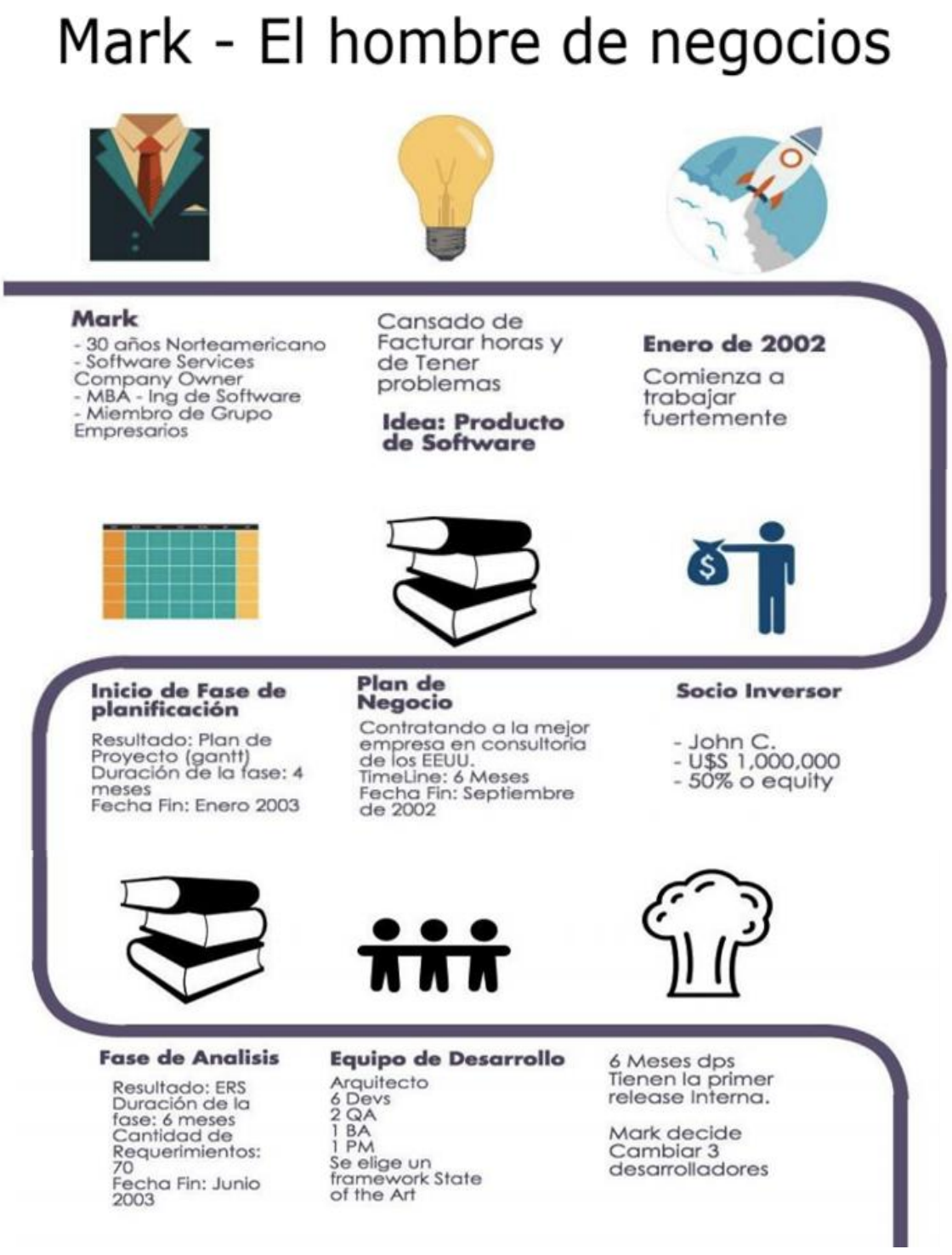
7) On the other hand, these methodologies allow quickly noticing both errors and problems. In the planning stage, the team presents a roadmap anticipating and responding to major technical issues and the speed at which they can work. With more traditional methodologies, unidentified errors in the early stages of the project often lead to very high costs.

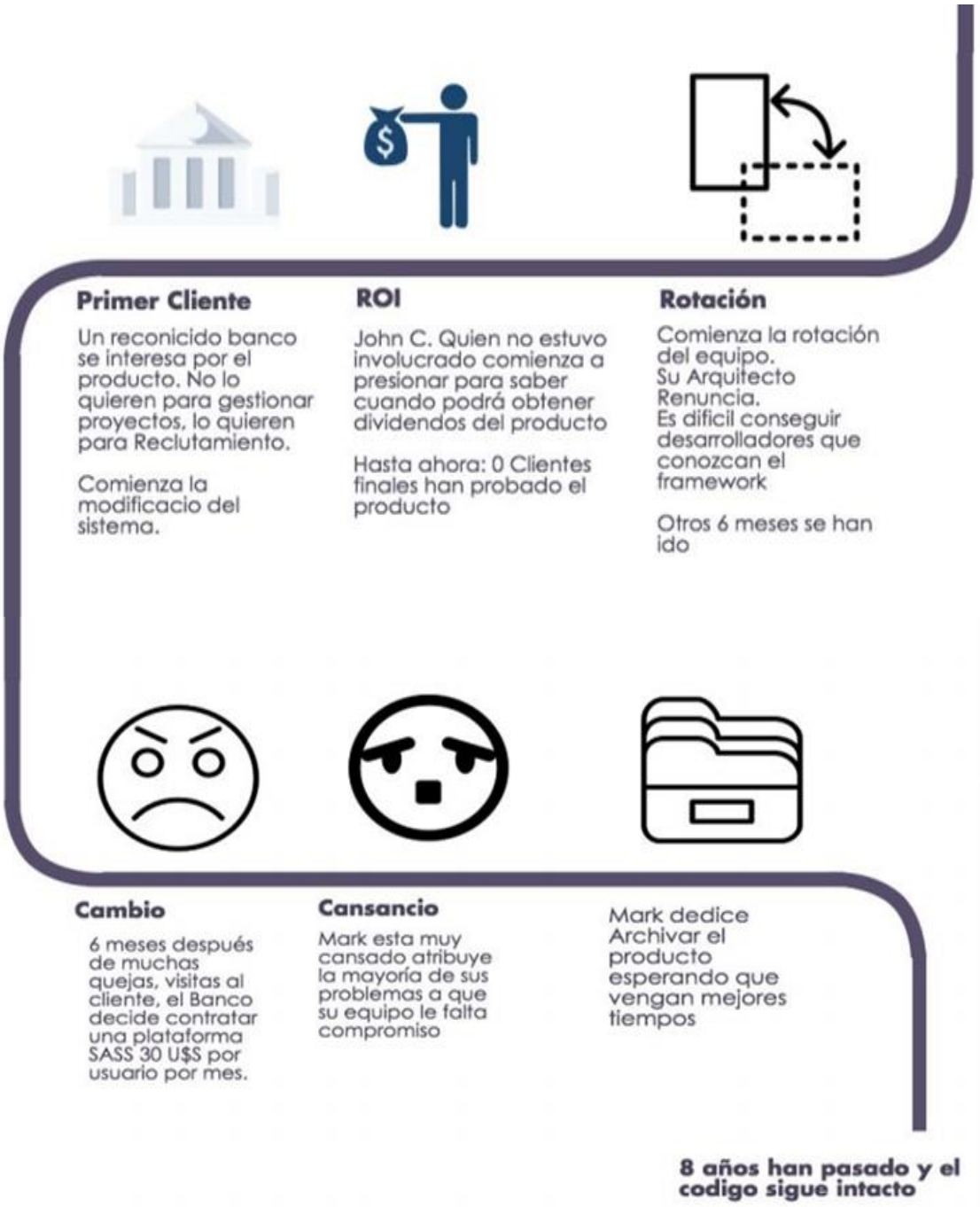
8) And finally, agile methodologies allow a faster return on investment. Thanks to the implementation of early deliveries, the client will have quick access to those functionalities that truly add value by accelerating the return on investment. (Martínez, 2014, https://goo.gl/a4kgpG).

4.1.1 Management Styles

Below, the following continuum shows two stories about two management styles that can be found in companies and ventures:

Figure 1: Entrepreneurial stories





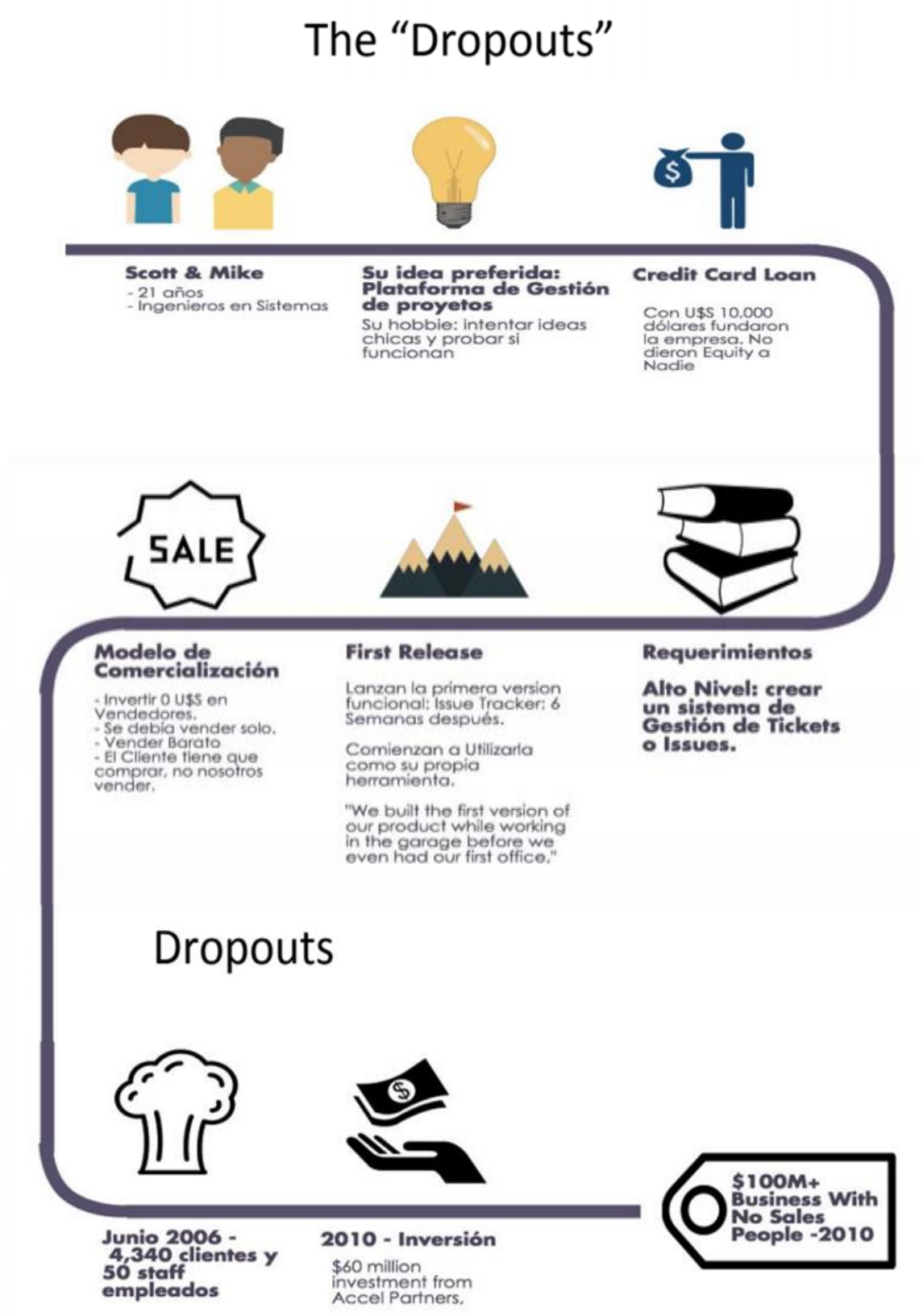
Source: Incutex, 2017.

The English translation of this image is below.

Mark - El hombre de negocios	Mark – The business man
Mark -30 años Norteamericano -Software Services Company Owner -MBA – Ing de Software -Miembro de Grupo Empresarios	**Mark** Mark -30 years old, North American -Software Services Company Owner -MBA – Software Engineering

	-Member of *Grupo Empresarios*
Cansado de facturar horas y de tener problemas **Idea: Producto de software**	Tired of billable hours and problems **Idea: Software Product**
Enero de 2002 Comienza a trabajar fuertemente	**January 2002** He starts working hard.
Socio Inversor -John C. -U$S 1.000.000 -50 % o equity	**Investment Partner** -John C. -U$S 1,000,000 -50% of equity
Plan de negocio Contratando a la mejor empresa en consultoria de los EEUU. TimeLine: 6 meses Fecha fin. Septiembre 2002	**Business Plan** He hires the best consulting company in the U. S. Timeline: 6 months End date: September 2002
Inicio de fase de planificación Resultado: Plan de proyecto (gantt) Duración de la fase: 4 meses. Fecha fin: Enero 2003	**Beginning of Planning Stage** Result: Project Plan (gantt) Duration of the stage: 4 months End date: January 2003
Fase de análisis Resultado: ERS Duración de la fase: 6 meses Requerimientos: 70 Fecha fin: Junio 2003	**Analysis Stage** Result: ERS Duration of the stage: 6 months Requirements: End date: June 2003
Equipo de desarrollo Arquitecto 6 Devs 2 QA 1 BA 1 PM Se elige un framework State of the Art	**Development Team** Architect 6 Devs 2 QA 1 BA 1 PM A State of the Art framework is chosen
6 meses dps tienen la primer release interna. Mark decide cambiar 3 desarrolladores.	6 months after, they have the first internal release. Mark decides to change 3 developers.
Rotación Comienza la rotación del equipo. Su arquitecto renuncia. Es difícil conseguir desarrolladores que conozcan el framework. Otros 6 meses se han ido.	**Rotation** The team rotation begins. His architect quits. It is difficult to find developers who know the framework. Another 6 months have gone by.
ROI John C. quien no estuvo involucrado comienza a presionar para saber cuando podrá obtener dividendos del producto. Hasta ahora: 0 clientes finales han probado del producto.	**ROI** John C., who was not involved, starts putting pressure on knowing when he is going to earn dividends from the product. So far: 0 final clients have tried the product.
Primer cliente	**First client**

Un reconocido banco se interesa por el producto. No lo quiere para gestionar proyectos, lo quieren para reclutamiento. Comienza la modificación del sistema.	A well-known bank shows interest in the product. They don't want it for project management, they want it for recruitment. The modification of the system begins.
Cambio 6 meses despues de muchas quejas, visitas al cliente, el Banco decide contratar una plataaforma SASS 30 U$S por usuario por mes.	**Change** 6 months after many complaints and customer visits, the Bank decides to hire a SASS platform U$S 30 per user per month
Cansancio Mark está muy cansado, atribuye la mayoría de sus problemas a que su equipo le falta compromiso.	**Tiredness** Mark is very tired. He attributes most of his problems to his team's lack of commitment.
Mark decide archivar el producto esperando que vengan mejores tiempos.	Mark decides to archive the product, waiting for some better times.
8 años han pasado y el código sigue intacto.	**8 years have passed and the code is still intact.**



Source: Incutex, 2017.

Note: Dropouts are defined as students leaving college before they have finished their studies.

The English translation of this image is below.

The "Dropouts"	The "Dropouts"
Scott & Mike -21 años -Ingenieros en Sistemas	**Scott & Mike** -21 years old -System Engineers
Su idea preferida: Plataforma de Gestión de proyectos Su hobbie: intentar ideas chicas y probar si funcionan.	**Their favourite idea: Project Management Platform** Their hobby: trying out if small ideas work
Credit Card Loan Con U$S 10.000 dólares fundaron la empresa. No dieron equity a nadie.	**Credit Card Loan** With U$S 10,000, they founded the company. They did not give equity to anyone.
Requerimientos **Alto nivel: crear un sistema de Gestión de Tickets o Issues.**	**Requirements** **High level: creating a Ticket or Issues Management system.**
First Release Lanzan la primera version functional: Issue Tracker: 6 semanas después. Comienzan a utilizarla como su propia herramienta. "We built the first version of our product while working in the garage before we ever had our first office."	**First Release** The first functional version is launched: Issue Tracker. 6 weeks later. They start using it as their own tool. "We built the first version of our product while working in the garage before we ever had our first office."
Modelo de Comercialización -Invertir 0 U$S en vendedores. -Se debía vender solo. -Vender barato. -El cliente tiene que comprar, no nosotros vender.	Marketing Model -Investing U$S 0 in sellers. -It should sell itself. -Selling cheap. -The customer has to buy, it is not on us.
JUNIO 2006 – 4.340 clientes y 50 staff empleados	**JUNE 2006 - 4,340 clients and 50 staff employees**
2010 – Inversión $60 millon investment from Accel Partners.	**2010 – Investment** $60 millon investment from Accel Partners.
$100 M+ Bussiness With No Sales People - 2010	**$100 M+ Bussiness With No Sales People - 2010**

Let us review, then, the differences between the two perspectives of development:

Table 1: Differences in management styles

Mark - The business man	Dropouts
• We need a detailed business plan. • Let's not touch a line of code until we agree on what needs to be done. • Let's make a very good process to optimize teamwork. • We need the best sellers. • Let's not let them steal our product or idea! • This is due to a lack of commitment.	• They prioritize the construction of prototypes and MVP's. • They look to model a business using lightweight tools and not make a big business plan before they start. • They focus on validating the ideas and not on doing big market research. • Doing what is necessary at the minimum cost. • Visibility. • Self-directed teams. • Choosing people wisely.

Source: Own elaboration, 2020.

These two stories indicate that there are two different management styles that can be found:

1. **Traditional**

- The planning style is longer. The planning is done months in advance. Projects have a long duration.
- Plans are more rigid.
- The boss has the fundamental role of controlling the team and the processes.
- The product specifications are detailed and long.
- The team follows orders.
- The client is not part of the team.
- Change generates negotiation and friction.

2. **Agile**

- There are long-term plans, but with permanent deliverables.
- The team is self-directed. The boss is a facilitator and not a controller.
- The specifications are detailed, but the prototypes are more important.
- The team participates in the decisions; it proposes alternatives.
- The client is part of the team.
- Change is part of the working style.

4.1.2 Agile Manifesto

All this spring from an agile manifesto written at the beginning of 2000 by a group of engineers who analysed the characteristics of projects that go well, identifying four main features:

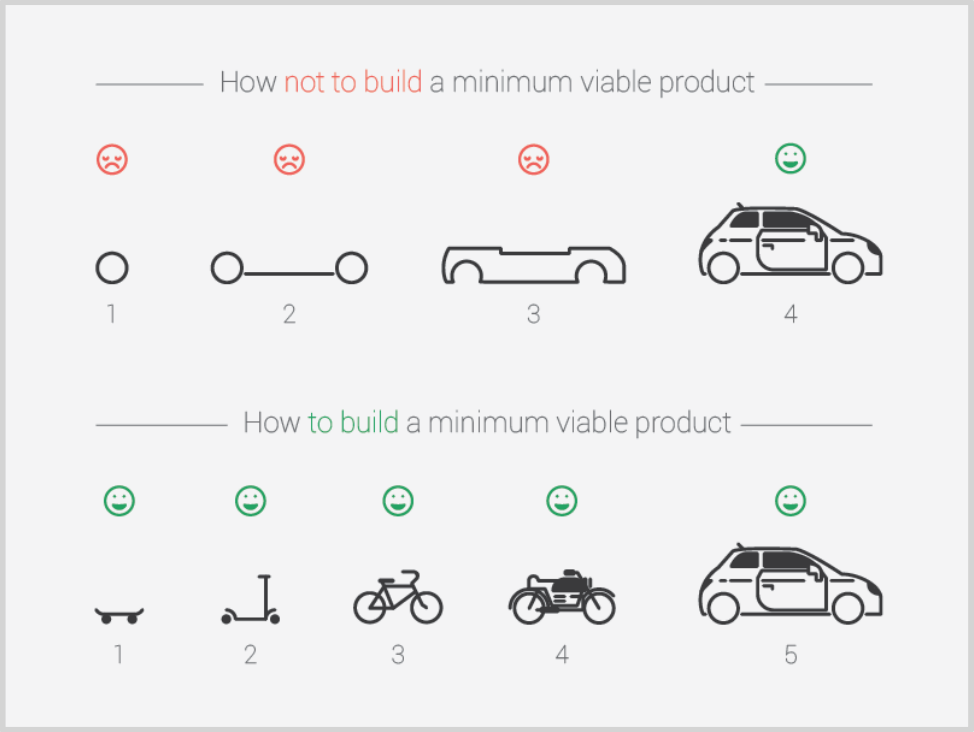
1. **Individuals and interactions over processes and tools:**
   We can have the best tools and processes, but if the team is not the right one or if they get along badly, that project will not move forward. It is more important to work on people and how they interact with each other than to work on processes
   This does not mean that the process and tools do not need to be defined, but the most important is to work on the team.
2. **Working PRODUCT / SERVICE over comprehensive documentation:**
   Prototypes are more valuable than extensive documentation. Here appears the concept of analysis paralysis, in which only analysis is done and thus the execution does not take place.
3. **Customer collaboration over contract negotiation:**
   When having long plans, many projects generate friction with the customer or supplier, internal or external.
   In an Agile project, collaboration is prioritized, many times yielding to contractual agreements.
4. **Responding to change over following a plan:**
   This point has to do with not conforming to issues defined in the plan and having the flexibility to change issues if they are no longer valid. (Beck et al., 2000, https://goo.gl/JY4H6D).

Let's consider the following example:

Spotify

It looks like a simple application, but it is a giant work of engineering with many modules and versions. One of its mentors is the one who signed the Agile Manifesto. Everything Spotify does about product construction is guided by agile methodologies. This person made the following graphic.

Figure 2: Spotify's Agile Approach



Source: Help Scout (n.d.) Retrieved from https://goo.gl/uTk4pc

At the top, you can see how products are **not** built in Spotify. For example, a customer shows up and tells them he wants a car. They take the order and tell him to come back in a month.

When the customer returns, they tell him that they started making the wheel. "Do you like it? Well, next month we'll show you another advance." When the client returns, they show him two wheel, and so on. After a month, he returns to see the wheels with the chassis and, in the fourth month, he will be able to see the finished car.
Many companies work with this traditional methodology approach.

In the case Spotify, when a customer orders a car, the first thing they ask is what he wants it for. When he tells them he needs it as a means of transportation, they set a date for to see the progress after two weeks. After two weeks, the client returns and is offered a scooter, with the aim of delivering something quickly to meet his need for mobility. And that way, the vehicle is improved in each iteration and, at the end, the same product is delivered.

The radical difference of this is that, in each iteration, the customer receives something of value, an incremental value. Providing the client, at each stage, with something concrete that can be touched and tested is fundamental in agile methodologies. This is the concept of delivering *prototypes to validate*.

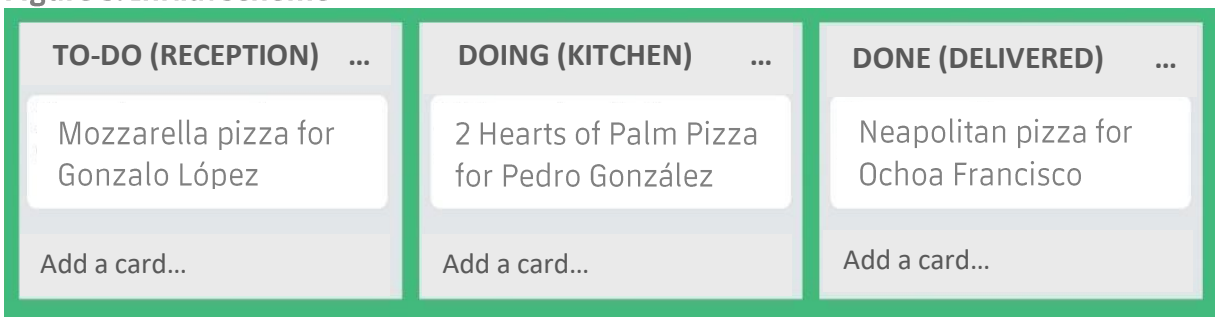
Next, we will develop two of the most important agile methodologies: Kanban and Scrum.

4.1.3 Kanban Methodology

Originally developed by Toyota, the Kanban methodology revolutionized the way companies maximize production efficiency. This consists of a very light logistical tool to produce things efficiently, based on what is needed.

One of its great advantages lies in the simplicity of the method. It is especially recommended for areas or projects in which there are many inputs and outputs and little planning. Let's look at a concrete example linked to the production of pizzas:

Figure 3: Initial scheme

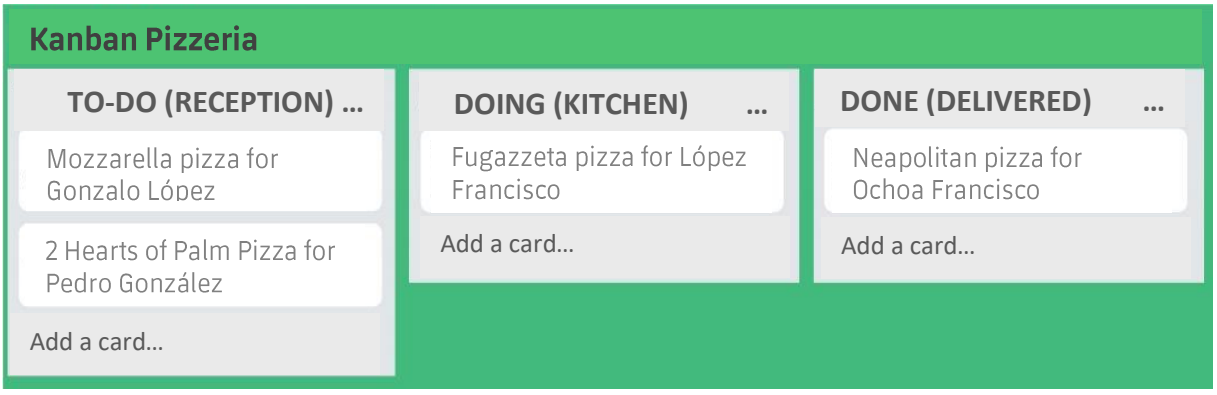


Source: Own elaboration, 2020.

Cards are used to plan and track production from reception to inventory and delivery. Then, a board must be done containing all the phases of the process:

- **TO DO: list of things to do.**
- **DOING: things that are being done or processed.**
- **DONE: things that are done.**

Figure 4: Board model



Source: Own elaboration, 2020.

Here appears an important concept which is the push versus the pull:

- **PUSH**
- **PULL**

Figure 5: Last pull phase. Card from doing to done

Source: Own elaboration, 2020.

- **As ingredients are needed, they are ordered (only the necessary ones).**
- **There is a clear visibility of what is being done at all times.**

When the following areas take the cards left by the previous areas, they are pulling. In contrast, in a traditional methodology, the boss or client would be who gives the card.

The main difference, then, is that when pulling, the person becomes independent of their work and there are no resources on hold. At the beginning of the day, the task list is presented and the team knows how to run it.

With push, at the beginning of the day, instructions are given to the team.

4.1.4 Scrum Methodology

Scrum is an agile approach to project management. It uses iterative and incremental processes, it is focused on results and commitments and it also has a very simple implementation.

As regards methodology, a planning time period is chosen; for example, 2 weeks. This would be a 2 week sprint planning. A project may have 20 sprints, but the planning horizon is 2 weeks.

There begins a series of project management activities:

1) The first thing is to hold a meeting where the tasks that are required to fulfil the objective are broken down, which is called a backlog.
2) On the first day of the week, a plan is made for those 2 weeks and activities are scheduled. This task is called grooming.
3) Once the activities and priorities have been defined, the team gets to work.
4) The team holds 15 minute meetings, every day early in the morning (or at some point during the day when all the members are able to meet; it is important that it

always takes place at the same time so as to generate the habit), which is called daily or stand up meeting, where the members answer three questions:

1. What did you do yesterday?
2. What are you going to do today?
3. Do you have any problem?

As the days and the two weeks go by, the tracking board is updated.

It is crucial that these meetings are held with all the participants standing (hence stand up meeting) so that they last less than 15 minutes. This way no one gets comfortable and stretches the meeting.

It is very important to be concise in these meetings and to respond briefly and specifically to each of the three questions. Any problems that arise are dealt with after or outside of that meeting in order to respect the 15 minutes. Moreover, that problem is dealt with only by the people involved, so the rest of the team is not held back or delayed.

The participants of the daily or stand up meetings are the people in charge and members of the team that is carrying out the project.

5) After two weeks, the demo or sprint planning and the retrospective take place:

- For the **demo**, the team meets with the client, shows all the work done, receives feedback and identifies the adjustments that need to be made. This meeting or event can last 1 hour. It is recommended that the client participates, as he is the one who will see the progress and give feedback.

- The **retrospective** is done after the demo. It is a meeting where you work with a board in which define the following:

Table 2: Review table

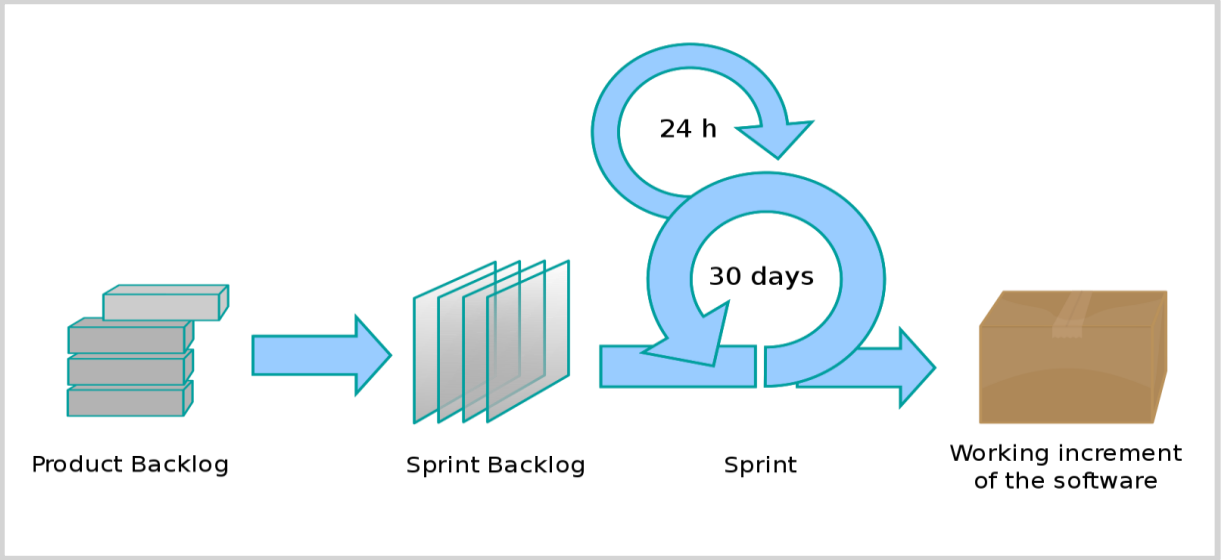
Things that went well	Things that went wrong	Things to improve

Source: Own elaboration, 2020.

In this meeting, the client may or may not participate. This activity can last up to an hour as well, but it is very important to generate the instance of learning and continuous improvement.

As a review, let's look at the Scrum life cycle:

Figure 6: Scrum Life Cycle



Source: Lakeworks, 2009, https://goo.gl/MijwBN

Roles and responsibilities in Scrum

1. Product owner:

- Defining the functionalities.
- Deciding on the delivery date and content.
- Prioritizing the product backlog.
- Being able to change the functionalities and priorities of each sprint.
- Accepting or rejecting the results of the sprint.
- Participating in sprint planning and review meetings.
- The product owner may be the same client (in many cases) or the team member who knows the client best.

2. Scrum master:

- Ensuring that the team is fully functional and productive.
- Promoting cooperation between all roles and removing barriers.
- Protecting the team from external interferences.
- Ensuring that the process is completed.
- Participating in the daily meetings, sprint planning and sprint review.

3. Team:

- Typically, 5 to 9 people.
- Selecting the sprint backlog.
- Having the right to do anything within the limits of the project guidelines to achieve the goal of the sprint.

- Organizing themselves and their work.
- Showing the results of the work to the product owner.

Let's look at some **keys to implementation:**

  - Adding value at each stage.
  - Separating the project into short phases and prioritizing requirements.
  - Learning from each stage.
  - Improving constantly.

To conclude, let's review **the benefits of this methodology:**

- No time is wasted in "non-productive" phases.
- Development begins early.
- Frequent deliveries.
- Early feedback.
- Changes are normal, they are not a problem.
- The focus is on business value, quality and time to market.
- Less paper! Only what is useful is documented.
- The client actively participates in all stages (product owner).

References

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunninghamm, W., Fowler, M..., & Thomas, D. (2000). Manifesto for Agile Software Development. Retrieved from http://agilemanifesto.org/

Ciotti, G. (2016). Why a Visual Really is Worth 1,000 Words. Retrieved from https://www.helpscout.net/blog/visual-communication/

Lakeworks. (2009). Scrum process [Image.] Retrieved from https://commons.wikimedia.org/wiki/File:Scrum_process.svg

Martínez, E. (2014). *Las 8 grandes ventajas de las metodologías ágiles.* [The 8 Great Advantages of Agile Methodologies.] Retrieved from https://www.iebschool.com/blog/que-es-agile-agile-scrum/