**Agile With Atlassian JIRA**

**Agile Approach:**

Agile is an approach to managing and working on projects that combines project management and product development.

It is a simple approach to managing complexity and is seen as an alternative to complex project management and heavy upfront planning.

Agile can be applied to any type of project, but it is most commonly used in software development.

Agile techniques can also be used in various departments within a company, such as human resources and marketing.

**Characteristics of Agile Projects:**

Agile projects are built incrementally, meaning that small valuable increments of the product are planned and released successively.

They are iterative, allowing for continuous feedback, learning, and improvement of both the product and the process.

Agile projects relentlessly focus on value, always working on the highest value parts of the project based on feedback.

They have an empowered team where members make decisions collectively and have the most current knowledge to make informed choices.

**Benefits of Agile Projects:**

From the customer's perspective, Agile projects deliver a desirable product by consistently incorporating customer feedback and delivering high-priority features early.

Agile projects lead to higher quality by continuously addressing mistakes and issues as they arise.

They foster higher job satisfaction among team members by empowering them to make decisions and utilize their skills and creativity.

Agile projects encourage innovation by allowing for experimentation and quick testing of ideas with real users.

They result in lower costs by focusing on value and avoiding wasteful activities, as well as resolving problems promptly.

Agile projects are safer as they continuously receive feedback, reducing the risk of building low-value features.

They offer predictable deliveries by ensuring shippable increments, even if the number of features may vary.

**The Scientific Method and Agile:**

Agile projects are based on the scientific method, which involves formulating hypotheses, conducting experiments, observing results, and learning from them.

Agile projects apply this iterative learning loop not only for discovery but also for accomplishing the work of a project.

The scientific method serves as a formalized description of problem-solving, which is an integral part of project work.

Various concepts and processes, such as "plan, do, check, act" or "think, build, ship, tweak," are derivatives of the scientific method and are used in Agile projects.

**Comparing Agile to the Waterfall Approach:**

The traditional waterfall approach manages projects by developing the entire product in distinct phases without looping or iterations.

The waterfall approach is similar to traditional mass production and relies on large batch sizes and economies of scale.

Waterfall projects face several downsides, including the upfront plan often being wrong, building unnecessary features, underestimated time and complexity, inability to adapt to market changes, difficulty and costliness of change, creation of obsolete documents, delayed feedback, and limited flexibility.

**Usage of Waterfall Approach:**

The waterfall approach is still used in certain situations where setup costs for each phase are high or when the work is relatively predictable.

For example, in the past, setting up computers and running tests required significant time and expense, leading to batching of testing in waterfall projects.

However, as the setup costs for project phases decrease and the work becomes less predictable, Agile approaches become more favorable and the waterfall approach becomes outdated.

**Introdução ao Jira:**

O Jira é um software usado para ajudar a gerenciar, desenvolver e comunicar projetos.

Pode ser usado em projetos individuais ou em equipe, sejam eles complexos ou simples.

O Jira é flexível e pode ser adaptado aos processos ágeis ou ao modelo cascata de gerenciamento de projetos.

**Hierarquia do Jira:**

Níveis de hierarquia: aplicação, projeto e problema (issue).

A aplicação Jira contém vários projetos.

Um projeto contém um ou mais problemas relacionados ao trabalho.

Um problema é uma tarefa ou trabalho a ser realizado no projeto.

**Navegação no Jira:**

Os projetos podem ser visualizados no menu suspenso "Projects" na barra de navegação superior.

Também é possível acessar projetos recentes ou problemas recentes por meio da opção "Your work".

Dentro de um projeto, há uma barra lateral com links específicos para o projeto atual.

**Tipos de projetos:**

O Jira oferece dois tipos de projetos: classic e next-gen.

Projetos classic são mais tradicionais, com mais funcionalidades e configurações compartilhadas.

Projetos next-gen são mais recentes, oferecendo maior facilidade de criação e configuração por membros da equipe do projeto.

**Administração no Jira:**

Há diferentes níveis de administração: site, Jira e projeto.

Um administrador de site tem acesso a toda a plataforma.

Um administrador do Jira pode controlar a criação de projetos e configurações que se aplicam a vários projetos.

Um administrador de projeto pode configurar especificamente um projeto em particular.

**Configurações e personalização:**

O Jira é altamente configurável e possui diferentes opções de personalização.

As configurações podem ser acessadas por meio do ícone de engrenagem na barra de navegação superior.

Existem configurações gerais do Jira, configurações de projeto e configurações pessoais.

**Conta e configurações pessoais:**

Cada conta do Atlassian Cloud tem configurações pessoais que podem ser personalizadas.

As configurações pessoais se aplicam a todos os sites associados à conta.

As configurações relacionadas a um site específico do Jira podem ser acessadas por meio das configurações pessoais.

**Recursos adicionais e aprendizado contínuo:**

Existem muitos recursos disponíveis para aprender mais sobre o Jira. Links para a página inicial do Jira, documentação, fóruns de discussão, treinamentos e canal do YouTube do Atlassian são fornecidos.

Pesquisas na web também podem fornecer informações úteis sobre tópicos específicos do Jira.

**Principles of Visualizing Work**

Visualizing work is an important principle of Agile.

A to-do list is a simple tool that visually reminds you of the work that needs to be done and helps you focus.

Prioritizing work items on a to-do list is a way to manage and organize tasks.

Tracking progress by checking off completed work items is rewarding and provides a sense of accomplishment.

Visuali zing work using tools like boards helps in organizing, focusing, and managing the work of a team.

Boards can be physical (e.g., whiteboard with sticky notes) or digital (e.g., software-based like JIRA).

Boards allow everyone to see the current state of the project, making it transparent to both the project team and stakeholders.

Visualizing work helps identify problems or bottlenecks in the process, facilitating continuous improvement.

Visualizing work can be applied to personal projects to increase productivity and effectiveness.

**Boards and Workflows**

Boards are Agile tools used to visualize and manage the work of a team.

Boards can be task boards, project boards, kanban boards, or scrum boards, depending on the context.

Boards can be physical or software-based, with examples like a whiteboard or JIRA.

Boards consist of columns that represent different stages or statuses of work items.

Work items (issues) are moved across columns as the team progresses through the project.

Boards provide a two-dimensional representation of work, allowing work items to go through multiple steps before completion.

Workflows are used to model the processes involved in a project, breaking down work into a series of steps.

Workflows and boards are closely related, with the board visualizing the workflow.

Boards and workflows can be customized and configured to match the team's needs and changing workflow over time.

**Configuring Boards and Workflows in JIRA**

JIRA automatically creates boards when projects are created using Kanban or Scrum templates.

Multiple boards can be created for a project, and a board can contain issues from multiple projects.

Workflows are associated with JIRA projects and define the available statuses for issues.

Changing an issue's status on a board involves moving it between columns, which represents a transition.

Boards can be configured to match the team's workflow by adding, removing, renaming, or modifying columns.

Column categories (to-do, in progress, done) help identify the status of an issue in its life cycle.

JIRA's simplified workflow is the default, but customized workflows can be managed to enforce step-by-step movement of issues.

Board cards display a small number of field values for an issue, such as type, priority, assignee, and additional custom fields.

Card layout and colors can be configured in JIRA to display relevant information and improve visual representation.

**Kanban Method Overview**

Agile methods are approaches used to achieve agility in project management. They are sometimes referred to as frameworks or methodologies. Agile is more of a mindset than a specific project management approach. Agile methods provide some structure to the agile ideas. Kanban and Scrum are common agile methods, along with others like Extreme Programming (XP). Although these methods differ, they all embody the core principles of agility, such as empowering the team, continuous improvement, working in small batches, and delivering value. Teams often combine ideas from different methods to create a customized approach for their specific needs.

The Kanban method is an agile approach commonly used to manage a continuous queue of work items or issues. It draws on ideas from the Toyota production system. Some key concepts in the Kanban method include limiting work in progress, ensuring the team only takes on a sustainable amount of work at a given time. It also emphasizes removing bottlenecks to improve workflow. While the goal is to achieve a steady flow of work, bottlenecks can occur due to process problems or the complexity of the issues. The team collaborates to identify and eliminate these bottlenecks, gaining a holistic view of the entire process. Addressing bottlenecks involves identifying the root cause and making process improvements.

The Kanban method promotes a "pull" approach to work, where individuals responsible for the next step in the process pull work from the previous step when they are ready, instead of having it pushed onto them. This helps maintain a smooth workflow.

**Why would a team choose to use the Kanban method?**

Kanban is a lightweight and efficient agile method. Compared to other methods like Scrum, it is even lighter and simpler. It provides a bare-bones approach to achieving agility, making it easy to understand and start using. Some teams find Kanban more effective than other agile methods. Additionally, Kanban can be adopted as an evolutionary approach to transitioning to agility. It allows teams to utilize existing roles and processes without requiring extensive reorganization or the introduction of new meetings or roles. Teams can begin implementing Kanban immediately and continually improve their agility over time.

The Kanban method works well for service-oriented workflows. It can be applied to various areas such as operations teamwork, support requests, maintenance development, and human resource departments' new hire processes. Anywhere there is a continuous flow of work can benefit from the Kanban method. However, Kanban can also be used for product development. It supports multi-team and multi-project workflows, allowing issues to be moved among teams using a single board or dedicated boards for each team. Each team has the flexibility to handle the work in their own preferred way.

**Kanban Boards:**

Like other boards, a Kanban board is a tool used to visualize a team's workflow. The default Kanban board in Jira consists of four columns: Backlog, Selected for Development, In Progress, and Done. When issues are created, they automatically appear in the Backlog column. Issues in the backlog may require further discussion or planning before they are ready to be worked on. When an issue is ready, it is moved to the Selected for Development column, which represents the team's prioritized to-do list. When a team member is ready to work on an issue, they pull it from the Selected for Development column into the In Progress column. Once the work on an issue is completed, it is moved to the Done column.

Kanban boards provide a simple and effective way to visually manage a team's work. While the default configuration is shown, the column layout can be customized to match the team's specific needs. Kanban boards focus on improving the flow of issues through the workflow. New issues continuously enter the backlog and are prioritized, usually by the business team. Once ready, they move through the workflow statuses. Work is completed before new work is started, ensuring the team always works on the most important issues as defined by the team.

**Separating the Kanban Backlog from the Kanban Board:**

In Jira, the backlog column on a Kanban board can be separated from the rest of the board, offering several advantages. It allows the development team to focus on issues they can work on, as backlog column issues are not ready yet. A separate backlog column is easier to manage, and its contents are not visible to the rest of the team.

To move a column from the Kanban board to the Kanban backlog, you can access the board's settings, go to the Columns tab, and drag the desired status or statuses to the Kanban backlog section. Once configured, the Kanban board no longer displays the backlog column. Instead, Jira adds a backlog tab to the sidebar, providing visibility and management of the backlog. The issues in the backlog are presented as a list, allowing for easier editing. Above the backlog, the first column of the board facilitates moving issues from the backlog onto the board, enabling the team to view and work on them. Modifying an issue's status from the backlog causes it to move out of the backlog and onto the Kanban board.

Kanban is a lightweight agile method that utilizes Kanban boards to manage the continuous flow of issues from backlog to done. In Jira, the Kanban backlog can be separated from the Kanban board, simplifying the board, and allowing separate backlog management.

**Work in Progress Limits (WIP):**

Limiting the amount of work in progress (WIP) on a Kanban board has several advantages. By setting minimum and/or maximum limits for the number of issues in specific columns, the flow of work improves. This means that fewer issues are being worked on simultaneously, resulting in a smoother and more efficient workflow. When work in progress is limited, the team focuses on completing the existing work before starting new issues. This approach reduces multitasking, which can be detrimental to productivity.

One of the benefits of limiting work in progress is faster issue delivery. With fewer issues in progress, they are less likely to pile up in certain columns, waiting for work to be restarted. As a result, the team can deliver completed work to the customer more promptly. Moving issues into the "done" status becomes a primary measure of the team's progress.

Moreover, limiting work in progress helps quickly identify any bottlenecks in the process. Since there are relatively few issues in progress at any given time, any problems or obstacles become more visible and can be addressed promptly. This continuous improvement of the team's work contributes to enhanced efficiency.

Another advantage of work in progress limits is the reduction of waste in the process. When there is a backlog or accumulation of issues in a particular status, it indicates a delay in working on those issues, resulting in wasted time and effort. Limiting work in progress ensures a more streamlined workflow and minimizes the need for rework caused by problems in earlier steps.

Additionally, work in progress limits foster teamwork. By restricting the number of issues in a particular status, team members are encouraged to collaborate and resolve any blockages together. This promotes a sense of shared responsibility and a collective effort to clear any obstacles.

To set up work in progress limits for a column on a Kanban board, access the Columns tab in the board settings. In Jira, these limits are referred to as Column constraints. You can define constraints based on the number of issues or exclude sub-tasks from the count. The column constraints are visible below the column name, indicating the current minimum and maximum constraints.

Determining the appropriate work in progress limits depends on the specific project and team dynamics. It is recommended to start without any limits initially and observe how the flow of issues progresses. If issues are not flowing smoothly or if there are instances where there are no items in a particular column, it might be necessary to introduce minimum constraints to ensure a continuous flow of work. Work in progress limits can also be set to discourage multitasking, such as having a limit of one issue per team member at a time. Additionally, limits can be placed on specific steps that tend to be neglected to ensure the team's attention is distributed evenly.

**Pulling versus Pushing work:**

In a workflow, work can be either pulled or pushed depending on the step and availability of resources. To illustrate this, let's consider an example of a restaurant taking and delivering orders. In some cases, the work is pushed forward, while in others, it is pulled.

For instance, after the wait staff takes an order, they usually push it to a queue, where it waits until they have the opportunity to start processing it. Once the cook becomes available, they pull the order from the queue and begin preparing it. After the food is ready, the cook pushes the order to the wait staff delivery queue. Finally, when the wait staff is ready, they pull the order from the delivery queue and deliver it to the customer:

Diagrama

Descrição gerada automaticamente

In this workflow, we can observe both push and pull mechanisms in action. The overall process follows a pull system since the cook only prepares orders upon receiving them. If there are no customers, the cook doesn't prepare any orders, thereby avoiding wastage. In contrast, a push system would involve preparing the orders ahead of time and pushing them onto the wait staff, even if there are no customers waiting.

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Kanban reports and metrics:**

Kanban provides various reports and metrics to visualize project status, troubleshoot issues, and plan for improvements. Two important metrics to track are lead time and cycle time. Lead time measures the time it takes for an issue to move from the request stage to the completed stage. Cycle time, on the other hand, measures the actual time spent working on an issue.

Kanban boards can generate reports like cumulative flow diagrams and cycle time control charts. Cumulative flow diagrams show the number of issues in each column over time, highlighting any bottlenecks or areas of concern. Cycle time control charts depict the average cycle time for issues over a specific period, allowing teams to monitor their performance and identify opportunities for optimization.

Gráfico

Descrição gerada automaticamente

Linha do tempo

Descrição gerada automaticamente com confiança média

Jira, a popular project management tool, provides automatic real-time reporting to support Kanban efforts. These reports help teams assess their progress, identify areas for improvement, and make data-driven decisions to enhance their workflow.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

**SCRUM OVERVIEW**

According to the Scrum Guide, which is written by the creators of scrum. Scrum is a framework for developing, delivering, and sustaining complex projects. It is a relatively simple framework for dealing with complex, unpredictable projects. By framework, we mean that scrum contains basic structures and ideas for completing a project. The scrum guide refers to scrum as a process framework for your project management and work techniques rather than a standalone process or definitive method. So, the basic ideas of Scrum can be customized to suit your specific project. This contrasts with a rigid methodology in which every project is executed the same way. Much of what we discussed in this video is discussed in the Scrum Guide. It is free, relatively short, and very well-written. It's highly recommended that you read it. Scrum is a way of achieving the idea of agility. In the previous videos, we have seen that Kanban is another option. You can think of agile as a mindset and the methods and frameworks such as Kanban, Scrum, and XP as ways of achieving agility.

A key component of scrum and agile in general is continuous learning. Scrum projects start with a vision. This is the initial desired end result of the project. The stronger the vision at the start of the project, the better. The product is then work done and iteration at a time.

After the first iteration, we have some of the features of the product. Even though the product only has a few features, it can be considered usable with respect to those features. That product should have value and potentially can be given to the customer. After the first iteration, notice that our vision has slightly changed. This is because we have learned from implementing the first product features and adapted our vision accordingly.

After our second iteration, the product contains the work of the first iteration as well as the work of the second iteration. It now has more features that the customer will value. Because of continuous feedback, we may have improved some of the features from the first iteration. Notice that our vision has again been adapted.

After the third iteration, our product is closer to our vision, but our vision has again changed slightly.

You can see that we are building the project incrementally because we are building it piece by piece. We are building it iteratively because the product gets improved as we are learning along the way. That is why Scrum is both an incremental and iterative approach to building projects. This process of building and improving parts of the product continues indefinitely, allowing the product to stay relevant in a changing marketplace.

If the product that is being built has a physical aspect, such as a rocket, the result of an iteration may be a prototype rather than a part of a completed product. At the end of every iteration, a product called an increment is ready. Here we see that three iterations have created three increments of the product. An increment is a usable product that may be given to the customer.

The organization always has the option to release the increment. Each increment must meet the organization's agreed upon definition of done. This includes quality and security standards, as well as other organizational requirements, such as requiring documentation for each feature. Each increment contains the work of the current iteration as well as the work of all prior iterations. In other words, an increment is the complete state of the product after an iteration.

In scrum, iterations are called sprints. A sprint is a time-boxed to period used to work on an increment. The time period of a sprint is fixed. In general, you do not shorten or lengthen the duration of the current sprint. Sprints usually have a duration of 1-4 weeks, with two-week sprints being typical. Shorter sprints create an opportunity for more adaptation. Longer sprints allow for more work to be done in a single increment. It's up to each team to decide on the appropriate sprint length.

There are three main parts of the scrum framework:

Artifacts are tools that allow for transparency of the project. They allow anyone with access to them to see the current state of the project.

The artifacts that we will talk about are the product backlog, the sprint backlog, the sprint goal, the sprint board and the sprint reports.

The second part of the Scrum framework are the roles related to scrum. We will discuss the roles of product owner, scrum master, development team members and stakeholders.

The third main part of the Scrum framework are the events related to scrum. These are also called ceremonies or meetings. The sprint guide considers the sprint as a container event for other events. In We will discuss the sprint planning meeting, daily standups, the sprint review, and sprint retrospective.

**Scrum artifacts:**

The main purpose of the artifacts is to provide project transparency. Anyone with proper access can use the artifacts to see the current state of the project, including the project's history and future plans. This enables the team to have a shared understanding of the project so that everybody is on the same page. The scrum artifacts are used to enable inspection and adaptation both inside and outside of scrum meetings.

**Product Backlog** is an ordered, ever-changing to do list for the project. It contains issues that are not yet part of any sprint. The scrum guide refers to issues as items. You might also hear them referred to as stories. Constant feedback means that the product backlog is always changing. The product backlog can include issues that represent features, improvements, bug fixes or any other type of issue that you would like. The product backlog is ordered. Issues near the top of the backlog are the closest to being worked on, so they usually have more details than the lower items.

Modifying the product backlog is called product backlog refinement. You might also hear this referred to as backlog grooming. According to the scrum guide, each scrum team decides how to do refinement, but it should consume no more than 10 percent of the development team's time.

In Jira when you are ready to plan a sprint, you navigate to the product backlog and click the "Create Sprint" button. After clicking the Create Sprint button, Jira creates an empty sprint. You can see here that Jira named the sprint using the project key, which is PRJ in this example, and the sprint member which is one in our case. You can see that Jira invites you to drag issues from your product backlog into the sprint.

The list of issues to be completed during the sprint is called the sprint backlog. The sprint backlog includes a plan on how to accomplish the work of the issues. In Jira, this means that before starting the actual sprint, more details are added to the issues in the sprint backlog. Those details describe how the work of the issues will be done.

As part of planning for sprints, it is common to estimate how much work an issue will take. **Story points** are the most common estimation statistic. In Jira you can use story points, hours, issue count or create your own estimation statistic. Story points are a relative measure of the amount of work required to complete an issue. For example, an issue that is assigned two story points is assumed to take about twice as long to complete as an issue that is assigned one story point.

In Jira, there's a field on each issue named story points. In the sprint backlog, you can see that the story points are shown in the gray boxes along with a total estimate of three points for the sprint backlog. Story points are used to help the team decide how many issues can be completed in a sprint.

When you want to start a sprint, you click the "Start Sprint" button associated with the sprint. The start sprint screen appears.

Jira starts by reminding you that you have added issues to the sprint backlog. You can modify the sprint name, specify the sprint's duration, and specify the start date for this sprint.

You can see that you can have the sprints started at a later date, so you don't have to actually click the start sprint button on the first morning of the sprint. You could also set up multiple sprints at one time.

The start sprint window also contains a place to enter what is called the sprint goal. The sprint goal represents the objective of the sprint’s increments. The sprint goal is reached by completing the issues in the sprint backlog.

A scrum role is that the sprint goal does not change during the sprint. The sprint is considered a success if the sprint goal is reached.

There are two major reasons to have a **Sprint Goal**. The first is that it provides a coherence to the product increment. This means that the features are related so that the product increment is valuable rather than building a collection of unrelated features. This also results in the Scrumteam working together to achieve the Sprint Goal.

The second reason is that it enables flexibility with the sprint backlog. Projects are complex and even though the sprint duration is relatively short, the team cannot predict the future and will learn and adjust during the sprint. There must be flexibility somewhere.

The sprint goal remains fixed during the sprint, but the issues that achieve the sprint goal can be modified as long as quality is not decreased. This means that there's flexibility in the makeup of the sprint backlog as the sprint is worked on. The sprint goal provides guidance for decisions as the team makes the adjustments.

A sprint has a **sprint board**. Notice that it only contains the issues in the sprint backlog. Issues in the product backlog or issues that are assigned to other sprints are not shown on the sprint board. Even in sprint projects, boards are often called Kanban boards, so don't be confused if you hear that term related to a sprint.

The **reports** in agile are tool to visualize the work, promote transparency, help with troubleshooting and continuous improvement, and help with planning and estimating.

In Jira, you access reports using the Reports tab in the sidebar. You can see that Jira automatically provides many reports related to your project.

Scrum has some common reports related to sprints. We will discuss the Burndown chart, spring report, and velocity chart.

A **Burndown chart** shows the progress that the team makes during a sprint.

The sprint backlog starts with a certain number of issues, each with an associated number of story points or other estimation statistic. The total number of starting points is shown on the left of the chart. This is the number of story points that the development team estimated that it would complete in the sprint. In our case, this sprint has three story points. The gray guideline shown is used to show the number of story points that should remain on a given day, assuming a linear Burndown of story points. On the last day of the sprint, the guideline reaches zero story points. This means that the work of the sprint should be finished on the last day. Notice that the non-working days are shown in the chart and the guidelines assume no progress will be made on those days.

As the sprint is underway, Jira will automatically update the Burndown chart as the status of the issues are updated by the team. The red line shows the actual number of remaining story points over time. You can see that about two days into the sprint, one story point was completed. On the last day of the sprint the remaining two-story points were completed.

Consulting this chart is an easy way to see if the team is on track for the current sprint. If the red line is below the gray line, your team is on track to complete all of the story points and reach the sprint goal. If not, the team may need to make some adjustments to reach the sprint goal.

The **sprint report** contains a nice summary of the sprint. It shows the Burndown chart as well as the current status of all of the issues in the sprint. This is an easy way to see how the sprint is progressing.

**Velocity** represents the rate at which the team accomplishes work. Usually, it is the number of story points completed per sprint.

Some teams use an estimation statistics other than story points, so in that case, velocity measures some other units completed per sprint.

You can see the team's velocity of a single sprint by looking at the Burndown chart. In this sprint, the team completed three story points, so its velocity is three. The velocity chart shows the estimated and the actual velocity of the team over time.

Scrum is an agile framework. An increment is a potentially shippable portion of the project that meets the "definition of done". A sprint is a time boxed period in which the increment is created. Scrum artifacts provide project transparency, enable shared understanding, and enable inspection and adaptation. Artifacts include the product backlog, the sprint backlog, the sprint goal, sprint boards and reports. Velocity is the rate at which the team accomplishes work, usually in story points per sprint.

**Scrum roles:**

A scrum team is made up of three roles: **product owner, scrum master, and development team**. The scrum team is made up of cross-functional team members allowing it to complete stories within the team. It is flexible and adaptable with members willing to help out where needed and continuously learning new things.

It is self-organizing: The team is responsible for deciding how to organize and do it's work.

Stakeholders are not members of the scrum team but are interested in the success of the project. There are internal stakeholders such as company managers, executives or other scrum teams that rely on the work of the scrum team. There are also stakeholders that are external to the organization such as customers, partners, and investors.

The **product owner** is the member of the scrum team who is responsible for communicating the product vision. The stakeholders and the scrum team need to have an understanding of the product vision in order to work effectively. The product owner is also responsible for maximizing the value of each increment. Each feature that the development team works on should be of high value.

The product owner is responsible for the product backlog. Others may help with the product backlog, but the product owner is responsible for it. Stakeholders primarily interact with the product owner. The product owner represents the stakeholders when they are not part of the discussion such as during scrum team meetings. The product owner is accountable to the stakeholders for the success of the project.

The **scrum master** is the member of the scrum team who is primarily responsible for promoting and supporting scrum, for scrum team members as well as stakeholders. It is up to the scrum master to ensure that everyone understands how and why things are done a certain way. The scrum master is also responsible for the day-to-day effectiveness of the scrum team. This includes ensuring that the team is reaching its goals and continuously improving.

The scrum master is responsible for protecting the focus of the team. This may mean helping to remove bottlenecks or ensuring that those outsides are interacting with the team in helpful ways. In general, scrum masters do what they need to do to allow team members to focus on their work.

The scrum master is also responsible for increasing the transparency of the project. There should be no surprises about the current status of the project.

The scrum master's tasks vary by project and from day-to-day. Typical tasks include coaching the scrum team and stakeholders on scrum and agile, helping to remove blocking issues, facilitating scrum events, configuring scrum artifacts, and monitoring sprint progress. This distinction of the product owner and scrum master roles should now be pretty clear

The **product owner** is responsible for the value of the product. The **scrum master** is primarily responsible for the effectiveness of the team. You can think of this both as a divide and conquer approach because the work of the roles is so different. Also, as a checks and balances approach because combining the roles could put too much weight on one of the responsibilities.

The separate roles lead to greater team success and sustainability. The development team is a cross-functional, adaptive team that does the work of the project.

Responsibilities of the development team include:

* Estimating issues – this is usually done using story points, but other estimation methods can be used;
* Deciding on how much work can be done in the sprint – only the development team can decide how much work to take out. It is assumed that the development team is in the best position to forecast how much work the issues take;
* Deciding how to organize to do the work of the team – this is because the team is an empowered, self-organizing team;
* Creating the increment of each sprint;
* The development team are the only members allowed to modify the sprint backlog during the sprint – This is sometimes necessary as the team learns and adjusts as it is building.

The Scrum Guide recommends having from three to nine members of the development team. Fewer than three members decreases the productivity and quality created by a cross-functional group of people working together. More than nine members tends to increase the amount of coordination required.

Jeff Bezos at Amazon refers to two pizza teams, meaning two pizzas should be able to feed the entire team.

To scale scrum to more than nine people, it is usually better to create multiple teams.

**Scrum events:**

A sprint contains four types of **events**. You may also hear events referred to as **ceremonies** or simply **meetings**. The events are the **sprint planning meeting, the daily standups, the sprint review, and the sprint retrospective**.

Scrum events occur at regular intervals and minimize the need for other meetings. These events are designed to maximize the opportunities for feedback, and continuous learning, and are a key part of achieving agility. In between these meetings is where the work of the issues of the sprint is completed.

All the **scrum meetings** have some common characteristics:

* The meetings have a fixed maximum time limit and no minimum time limit;
* Meetings can never go over their allotted time, but they can be ended early if the purpose of the meeting is achieved;
* The meetings are primarily to plan, inspect, and adapt.

In agile projects, the planning is distributed throughout the project rather than the mostly upfront planning of waterfall projects. These meetings are an important part of that distributed planning. The meetings are used to inspect the project and adapt with the team is doing based on that inspection. This is a key part of increasing transparency and continuous improvement.

The meetings are primarily about the team collaborating, not about updating status. The work is visualized, so you don't need a meeting to see the status. It is the responsibility of every team member to ensure that meetings have value and to help modify them to increase their value if necessary. If the discussion moves in a direction that becomes a value to only a portion of the participants, it should be moved to a later time outside of the meeting.

The **sprint planning meeting** is held at the start of a sprint. The entire scrum team attends the meeting. The duration of the meeting is typically four hours for a two-week sprint. If your sprints are four weeks long, you can expect this meeting to be twice as long. The purpose of the meeting is to plan the work of the sprint. The output of the meeting is an agreed upon sprint goal and a sprint backlog.

Before the meeting, the product owner usually has a proposed sprint goal and a minimum set of issues that accomplish the goal. These preliminary items often come from the product backlog that has been updated during the previous sprints, sprint review, and sprint retrospective meetings.

During the meeting, the team usually discusses the sprint goal, modifies the sprint backlog, places story point estimates on issues, and adds details to the issues to better describe the specific work to be done.

The **development team** is responsible for estimating the story points for the work and deciding on how much work can be done during the sprint. They need to do enough planning to have an accurate forecast for the amount of work that they will agree to.

The **development team** also creates subtasks for the first few days of the sprint. The subtasks are often a day or less of work. This is an example of an empowered team rather than a command and control-based team.

The purpose of the meeting has been met when there is an agreed upon sprint goal and sprint backlog. The sprint backlog contains the issues that the development team has agreed to complete during the sprint, as well as a plan for completing that work.

The **daily standup meeting** is also called the **daily scrum**. It is a planning meeting that occurs every day and the participants usually stand as a reminder that it's a short meeting.

The meeting usually takes place in the same location at the same time every day. The development team are the primary attendees for the daily standup. Others may attend the meetings but are usually asked to listen only.

The meeting usually last 15 minutes or less. The purpose of the meeting is to inspect recent progress toward the sprint goal, plan the day's work, and identify any impediments to the team.

The team usually makes plans to resolve the impediments, but the discussion often moves to after the meeting.

The output of the meeting is the plan for the day.

It's important that the daily standup is a collaborative meeting and not simply a status update. The team collaborates to make the best decisions for the day given the latest information. They usually decide on who will work on specific issues, to plan slightly changes after every daily standup. This is continuous improvement.

The **sprint review meeting** occurs near the end of the sprint. It is an informal meeting that includes the scrum team and interested stakeholders.

It is typically a two-hour meeting for two-week sprints.

The purpose of the meeting is to inspect the increment that was just created in the sprint and to collaboratively update the product backlog. This is a meeting with a lot of feedback on the project and includes a brainstorming session to help decide what to do next.

The output of the meeting is a first pass at the next sprint's backlog. By the time the sprint planning meeting happens for the following sprint, the team already has a good idea of what they will be working on.

The **sprint retrospective** is the last event of the sprint. The scrum team attends the retrospective.

The meeting typically takes 90 minutes for a two week sprint.

The purpose of the meeting is for the team to inspect itself including its processes, tools, and team interaction.

The retrospective is a positive meeting containing constructive feedback. Everyone should always remember that they are part of a team.

The scrum master usually helps make sure that this is a positive meeting. The team usually discusses what they should keep doing, what they should stop doing, and what they should start doing. The meeting is about continuously improving the team.

The output of the meeting is to add one or more improvement related issues to the next sprint's backlog. It's important that the team spent some of it's time on these issues, rather than exclusively building the product.

It's important to always make sure that the meetings have high value, and the team should focus on continuously improving their meetings.

Notice that the next sprint goal and sprint backlog start to form in the sprint review.

The retrospective usually adds one or more issues. This forms the starting sprint backlog for the sprint planning meeting.

* Scrum roles include the product owner, scrum master, development team members, and stakeholders;
* Scrum meetings include the sprint planning meeting, daily standups, the sprint review, and the sprint retrospective.

**Toyota Kanban**

**Toyota Productions System:**

In 1978, Taiichi Ohno considered to be the father of the Toyota Production System, wrote a book about the thinking behind the system. In this book, he describes what is now sometimes called **lean thinking** or **lean management**.

We will see that this book contains many of the principles that are used in agile projects.

After World War II,Toyota had a goal to catch up with America in three years in the automobile industry. Japan's automobile industry was very small at the time compared to American companies like Ford and General Motors.

The main focus of the Toyota production system is to eliminate waste and

increase productivity.

Taiichi Ohno had heard that American workers were something like nine times more productive than Japanese workers and decided that there must be a lot of wasted effort. The result was a decades long focus on eliminating waste and continuously improving. This eventually resulted in Toyota becoming the largest automobile company in the world.

Whether or not the productivity numbers were true, that's what Toyota believed and helps explain their focus on eliminating waste. Toyota studied and embraced many of the production ideas from Ford but used a more agile approach to producing cars.

After World War II, resources were very scarce in Japan. And Toyota simply couldn't afford to hold a lot of inventory as was done in traditional mass production. Out of need, Toyota developed a leaner, more flexible approach to mass producing cars. We will see that this lean approach uses many of the same principles that are used today in agile projects.

In his book, Taiichi Ohno emphasized the need for the team to have a clear purpose. He said “I would like to emphasize that the Toyota production systems

was realized because there were always clear purposes and needs.”

Lean principles of the Toyota production system can be applied in many contexts. They do not apply only to lean manufacturing or production. In fact, Taiichi Ohno wrote about this in the preface to the English edition

of the book, which came out in 1988. He said the “Toyota production system is not just a production system. I am confident it will reveal its strength as a management system adapted to today's era of global markets and high level computerized information systems.”

**Toyota Kanban**

Kanban is a Japanese word used at Toyota to generically describe an object that controls the flow of work. The actual object can vary depending on the situation, but it usually holds some information related to the work. By naming this type of object a kanban, Toyota basically was adding a new definition to the word.

The idea came to Toyota from the way that supermarkets manage their inventory.

Taiichi Ohno mentions in his book that he once came to the US to visit car companies but he was more excited about visiting and learning from the supermarkets.

In supermarkets, for some products, instead of forecasting and having a product delivered at regular intervals, which is a push system, more products are only ordered when the current inventory reaches a **reorder point**. **This is a pull system**. This matches the supply and demand.

For example, the supermarket doesn't need to guess how many cans of tomatoes to order or risk having too little or too much inventory. They can **wait for one of the boxes to empty**, then order another box. In this example, **an empty box is acting as a kanban**. **It signals employees to order more product**.

Taiichi Ohno said in his book, “from the supermarket, we got the idea of viewing the earlier process in a production line as a kind of store.” When a later process used a component such as an engine, it would provide a kanban to the earlier process. That kanban could be a piece of paper with information on it or

something like an empty cart.

This would signal the earlier process to build another component. The earlier process only builds a component when it has been given a kanban. This is sometimes called **adjusting time system**.

Even though the word **kanban** is used to represent work to do at Toyota, the concept has been around for a very long time.

Example: A guest check at a restaurant is a kanban.

* The work preparing an order does not start until the guest check is filled out;
* If no guests come in, no orders are prepared;
* The information related to the order is on the guest check.

An empty coffee cup when ordering coffee can also serve as a kanban. The size of the cup, your name, and the desired ingredients provide the information that is needed to start and fill your order.

This kanban also happens to be a convenient delivery vehicle for your order.

Another example of a kanban is a Jira issue. It provides information related to the work of the issue; without the Jira issue in the correct column, the work shouldn't be started.

You can see that all these kanbans represent and hold information related to work to be done.

Kanbans are not used in isolation. There is an associated system to them which sets the rules on how the kanbans are used.

Taichi Ohno said that the Toyota production system is the production method and the kanban system is the way that it is managed. He is pointing out that the kanbans are part of a kanban system.

Similarly, every restaurant uses guest check kanbans in slightly different ways, depending on the kanban system.

Also, agile projects that use issues as kanbans use them in unique ways.

There are many **benefits to using kanban** systems to manage work. Some of the benefits include:

* They **visualize the** work – kanbans are visual, allowing you to see very clearly what work needs to be done;
* They are simple – forecasting demand and managing inventories is usually a much more complicated approach to managing the work than using kanbans.
* They are reliable – this reliability comes from the visual aspect, and the simplicity of the system.
* They are efficient – an effective kanban system uses resources only to do work that is of value to the customer;
* They eliminate waste – instead of holding and managing inventory, the supply is matched to the demand;
* It is easy to identify bottlenecks or problems in the system and make adjusts for improving the workflow. This is mainly because the simplicity and just-in-time nature of the system makes it easy to see where problems occur.

The term kanban has a few meanings, depending on the context. Kanban may refer to an object that controls the flow of work. We have seen that restaurant guest checks and jira issues can be considered kanbans. We could call, these kanban, tokens to help clarify the meaning.

Kanban may also refer to the system that controls the flow of working

using kanbans.

We have seen that Toyota has a system for how it uses kanbans just as restaurants have a system for how they use gas checks and how a team uses jira issues. We could call this a kanban system to help clarify the meaning.

We have also seen that there is a lightweight agile method called kanban. It takes its name from the kanban system at Toyota but applies it to agile projects.

This is a bit confusing because kanban tokens, kanban systems and even things like kanban boards are used in most agile methods, not only in the kanban method.

**Review:**

* Lean principles apply broadly including on agile projects;
* A kanban is an object that controls the flow of work;
* A kanban system is a system that controls the flow of work using kanbans.

**Lean Principles**

With **Lean Principles**, in many ways, the principles are more important than the practices or tools. Principles tend to be foundational and long lived even as the practices and tools change.

Taiichi Ohno discusses the importance of understanding the principles in his Toyota Production Systems book. He said, "With a better tool, we can get wonderful results. But if we use it incorrectly, the tool can make things worse. We should not forget to **always use the principles**”.

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**

Toyota didn't invent many of these principles but the combination of them can and have been used very effectively for many types of projects and businesses.

One of the key principles of Lean is to empower and leverage the creativity and talent of the team. This is not only a smart business but improves overall team happiness.

Taiichi Ohno said, "Operators acquire a broad spectrum of production skills that I call manufacturing skills and participate in building up a total system in the production plant. In this way, the individual can find value on working."

These two sentences contain some key concepts related to the people on the team. First, they have a diverse set of skills including and understanding of Lean principles. This is different from a traditional mass production line or a command-and-control lead team, where team members tend to have specialized jobs. Acquiring this diverse set of skills means that the team members are always learning.

He also mentions that team members participate in building the system. This means that the team is trusted to make decisions and even have the ability to stop the production line, if necessary.

Finally, not only is empowering the team good for the success of the company, but it also improves team satisfaction.

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

Many people want to feel like they're contributing to something of value and empowering individuals provides that, even more than monetary or recognition rewards.

Teamwork is a big part of empowering the team.

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

Taiichi Ohno thought that we could learn a lot from sports teams. He mentions in his book that, “A championship team combines good teamwork with individual skill”. He stresses the importance of having a teamwork skill by saying, "In modern industry, harmony among people in a group, as in teamwork, is in greater demand than the art of the individual craftsmen”.

**Visualize Work**

Agile projects are usually visualized with boards and the progress of the team is visualized with charts. Toyota uses kanbans to signal and control the work.

Toyota also has something called an **andon board**. This is an information board that shows any existing problems. He describes this in his book, "When one looks up, the andon comes into view, showing the location and nature of trouble situations at a glance." He also said, "Look straight at the reality. "

An "andon board" is a visual management tool used in lean manufacturing or production environments to provide real-time information about the status and performance of the production line or process. It typically consists of a large display board or screen located in a prominent area of the workspace where it can be easily seen by the production team.

The purpose of an andon board is to provide **visibility** and transparency into the production process, allowing the team to monitor key performance indicators (**KPIs**) and quickly identify any issues or abnormalities that may arise. It serves as a communication tool that helps operators, supervisors, and managers stay informed about the current state of operations.

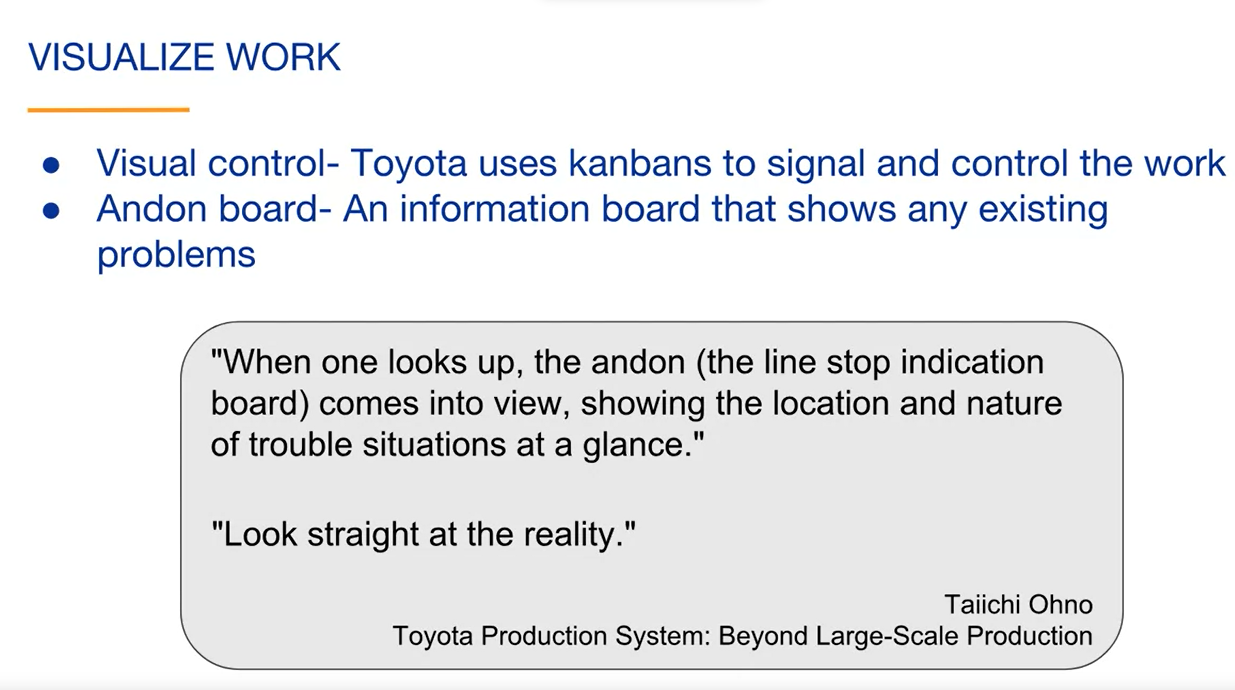
The andon board usually displays information such as:

* Production status: It provides real-time updates on production progress, including the number of units produced, production rates, and targets;
* Quality metrics: It shows information about product quality, including the number of defects, scrap rates, and first-pass yield;
* Performance indicators: It displays key performance metrics such as cycle time, downtime, and overall equipment effectiveness (OEE);
* Alerts and notifications: Highlights any abnormal conditions or problems that require attention, such as equipment breakdowns, material shortages, or quality issues.

By having this information displayed prominently, the andon board helps create a culture of continuous improvement and empowers the production team to respond quickly to any issues that arise. It promotes collaboration, problem-solving, and accountability among the team members.

Additionally, the andon board can be integrated with other systems or tools, such as production tracking software or sensors, to provide real-time data and enable automatic updates on the board. This ensures that the information displayed is accurate and up to date.

Overall, an andon board serves as a visual control tool that enhances communication, facilitates decision-making, and supports the overall goal of improving productivity, quality, and efficiency in the production environment.

****

Visualizing work tends to give a true transparent view into the status of the project, something that can otherwise easily be hidden.

**Scientific Method**

The scientific method or similar experimental evidence-based methods are used to continuously improve. The scientific method is the foundation of Agile methods, and this is also true with Lean thinking.

Here's a quote from Taiichi Ohno's book to support this, "The Toyota production system has been built on the practice and evolution of this scientific approach."

**Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente**

Related to continuously improving is the need to continuously learn. He said in his book, "Progress cannot be generated when we are satisfied with existing situations."

**Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente**

Every organization should be a learning organization. Every team should be learning team. Everyone should realize that even if they are successful now, they have a lot to learn. A big focus at Toyota has been to continuously improve.

Here's a quote from Taiichi Ohno was book that helps demonstrate that, "The new market demanded a constantly improving automobile." If you think about it, many markets such as markets for cell phones and software assume constantly improving products.

**Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente**

**A Lean principle is to embrace change** rather than create a fixed process and stick to it.

Here are some quotes from Taiichi Ohnos' book, "As long as we cannot accurately predict the future, our actions should change to suit changing situations. In industry, it's important to enable people to cope with change and think flexibly." "Build a fine-tuning mechanism into the business so that change will not be felt as change”.

**Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente**

The second quote points out that, in a lean or agile environment, change feels normal.

**Improving the flow of value**

In a lean environment, you want to set up processes so that the value that you are creating flows from start to finish. One of the main reasons to do this, besides faster delivery to the customer, is so you can immediately see and fix problems in your workflow. This leads to continuous improvement.

Taiichi Ohno discusses this by saying, "Just-in-time means that, in a flow process, the right parts needed in assembly reached the assembly line at the time that they are needed and only in the amount needed."

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

It's important to improve the whole system in addition to each part. He said, "Efficiency must be improved at each step and, at the same time, for the plant as a whole."

**Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente**

**Value stream mapping** is a tool used for continuous improvement. Toyota calls this material and information **Flow Mapping**.

* Draw diagram with the steps involved in the process that you want to improve. These often are the steps in bringing a product to the customer. This is called the current state;
* Identify the desired future state. This is done by finding ways to increase value and reduce waste in the process. This often includes finding better ways to do things, the organization's vision, and overall direction help to identify the desired future state;
* Iterate toward the future state. You may never fully reach it, but iterating towards it is a way to continuously improve.
  + **Obs.: "Iterate toward the future state"**, em **mapeamento do fluxo de valor**, refere-se ao processo de realizar repetidas iterações e melhorias no fluxo de valor atual, buscando alcançar o estado futuro desejado (conforme desenhado na etapa anterior). O estado futuro representa a visão ideal do fluxo de valor, onde os desperdícios são minimizados, o tempo de ciclo é reduzido e a eficiência é maximizada.
  + Ao mapear o fluxo de valor atual, a equipe **identifica os gargalos, os desperdícios e as ineficiências** presentes no processo. Em seguida, um plano é desenvolvido para alcançar o estado futuro desejado, que representa uma versão aprimorada do fluxo de valor.
  + No entanto, é importante reconhecer que a implementação de todas as melhorias de uma só vez pode ser desafiadora e impraticável. Portanto, o conceito de "iteração" é aplicado, o que significa que as melhorias são implementadas em etapas incrementais ao longo do tempo. **Cada iteração envolve a implementação de um conjunto de melhorias e, em seguida, a avaliação dos resultados obtidos**.
  + Ao "iterar em direção ao estado futuro", a equipe realiza sucessivas iterações, avaliando o impacto das melhorias implementadas em cada etapa. É feito um novo mapeamento do fluxo de valor para comparar o estado atual com o estado futuro após cada iteração, permitindo que a equipe identifique áreas que ainda precisam ser aprimoradas e identifique novas oportunidades de melhoria.
  + **O objetivo final é continuar aperfeiçoando o fluxo de valor ao longo do tempo, por meio de múltiplas iterações, até que o estado futuro desejado seja alcançado. É um processo contínuo de aprendizado, ajuste e aprimoramento, onde cada iteração contribui para a evolução do fluxo de valor em direção à visão idealizada.**

**Interface gráfica do usuário, Texto, Linha do tempo

Descrição gerada automaticamente**

**Value Stream Mapping (VSM)** is a fundamental tool in the Lean concept that focuses on improving processes and eliminating waste to maximize value for the customer. It is a visual representation and analysis technique used to map and understand the flow of materials, information, and activities involved in delivering a product or service.

The purpose of value stream mapping is to identify and eliminate non-value-added activities or waste within a process. By creating a visual representation of the current state of the value stream, it becomes easier to identify areas of inefficiency, bottlenecks, overproduction, unnecessary motion, waiting time, and defects.

The process of **Value Stream Mapping** typically involves the following steps:

* **Select the Value Stream**: Determine the specific process or area of focus for mapping;
* **Map the current state:** Create a visual representation of the current process, including all the steps, activities, and information flow from the beginning to the end. This includes capturing data on cycle times, lead times, and inventory levels;
* **Identify waste:** Analyze the current state map to identify waste and non-value-added activities. This can include excessive transportation, inventory, waiting, overprocessing, defects, and underutilized talent;
* **Design the future state:** Develop an ideal or future state map that eliminates or reduces waste and improves the overall flow of the value stream. This involves redesigning the process, resequencing activities, reducing lead times, and optimizing resource utilization;
* **Plan implementation:** Determine the necessary steps and actions to move from the current state to the future state. Prioritize improvement opportunities and create an action plan for implementation;
* **Implement improvements:** Execute the planned improvements, making changes to the process, based on the future state map;
* **Continuously improve:** Regularly assess the impact of the implemented improvements and iterate the value stream mapping process. This involves monitoring key performance indicators (KPIs), measuring the effectiveness of the changes, and identifying further improvement opportunities.

By visually representing the value stream and identifying areas of waste, value stream mapping helps organizations streamline processes, reduce lead times, improve quality, optimize resource utilization, and ultimately deliver higher value to customers. It is an essential tool in Lean manufacturing and continuous improvement efforts.

*O Mapeamento do Fluxo de Valor (MFV) é uma ferramenta fundamental no conceito Lean que se concentra em melhorar processos e eliminar desperdícios para maximizar o valor para o cliente. É uma técnica de representação visual e análise usada para mapear e entender o fluxo de materiais, informações e atividades envolvidas na entrega de um produto ou serviço.*

*O objetivo do mapeamento do fluxo de valor é identificar e eliminar atividades que não agregam valor ou desperdícios dentro de um processo. Ao criar uma representação visual do estado atual do fluxo de valor, torna-se mais fácil identificar áreas de ineficiência, gargalos, superprodução, movimentos desnecessários, tempo de espera e defeitos.*

*O processo de mapeamento do fluxo de valor geralmente envolve as seguintes etapas:*

*Selecionar o fluxo de valor: Determinar o processo ou área específica a ser mapeada.*

*Mapear o estado atual: Criar uma representação visual do processo atual, incluindo todas as etapas, atividades e fluxo de informações desde o início até o fim. Isso inclui a coleta de dados sobre tempos de ciclo, tempos de entrega e níveis de estoque.*

*Identificar desperdícios: Analisar o mapa do estado atual para identificar desperdícios e atividades que não agregam valor. Isso pode incluir transporte excessivo, estoques desnecessários, tempo de espera, superprocessamento, defeitos e subutilização de talentos.*

*Projetar o estado futuro: Desenvolver um mapa ideal ou do estado futuro que elimine ou reduza os desperdícios e melhore o fluxo geral do fluxo de valor. Isso envolve redesenhar o processo, reordenar atividades, reduzir tempos de entrega e otimizar a utilização de recursos.*

*Planejar a implementação: Determinar as etapas e ações necessárias para passar do estado atual para o estado futuro. Priorizar as oportunidades de melhoria e criar um plano de ação para a implementação.*

*Implementar melhorias: Executar as melhorias planejadas, fazendo mudanças no processo com base no mapa do estado futuro.*

*Melhorar continuamente: Avaliar regularmente o impacto das melhorias implementadas e iterar o processo de mapeamento do fluxo de valor. Isso envolve monitorar indicadores-chave de desempenho (KPIs), medir a eficácia das mudanças e identificar novas oportunidades de melhoria.*

*Ao representar visualmente o fluxo de valor e identificar áreas de desperdício, o mapeamento do fluxo de valor ajuda as organizações a otimizarem processos, reduzirem tempos de entrega, melhorarem a qualidade, otimizarem a utilização de recursos e, em última análise, entregarem maior valor aos clientes. É uma ferramenta essencial nos esforços de manufatura enxuta e melhoria contínua.*

You can see that this is a learning loop that is similar to the scientific method. It is also similar to the Agile approach of building the product incrementally.

Here are some quotes from Taiichi Ohno related to limiting work in progress. "Reducing the number of Kanban increases their sensitivity." In other words, if you limit the amount of work in progress, you are getting constant feedback and you tend to see problems easier because they don't get lost in a sea of other work. "People prefer working with large quantities. It's easier than having to work hard and learn from producing small quantities." In other words, some people like the comfort of not getting consistent feedback because it's easier to just continue what you are doing.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

We have discussed the concept of pulling work from the previous step rather than having work pushed on you. This allows you to limit inventory and manage the flow of value.

Taiichi Ohno discusses this, "The conventional way was to supply materials from an earlier process to a later process. So, I tried thinking about the transfer of materials in the reverse direction." He was a big proponent of using this kind of unconventional thinking to help solve problems. One of the most important principles in Lean is to eliminate waste

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Here are some quotes related to eliminating waste. "The basis of the Toyota Production System is the absolute elimination of waste." "All we are doing is looking at the timeline from the moment the customer gives us an order to the point when we collect the cash. And we are reducing that time by removing the non-value-added wastes."

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

"The vicious cycle of waste generating waste hides everywhere." He gives some examples of waste generating waste. For example, having too much inventory is a waste. It could cause the company to need a storage facility. This requires a system to manage the inventory. It also requires methods to make sure that the inventory does not degrade before it's used.

While this is a physical example, this concept of waste generating waste is true with other types of products. For example, having too many features planned in detail is a waste because some features may never be built.

Here are some types of waste that Taiichi Ohno pointed out in his book. These are examples from production at Toyota but with some minor changes, they can apply broadly.

Texto

Descrição gerada automaticamente com confiança média

Types of waste include overproduction or producing more than customers use, waiting to do or receive work, unnecessary transportation, waiting for the processing of information related to the product, holding unnecessary inventory, unnecessary movement of employees or product, and making defective products.

There are many other types of wastes depending on the circumstances. For example, if we think about how we use our time, most of us can identify wasteful activities that limit our productivity.

One way to reduce waste is to reduce setup times. Toyota has flexible factories allowing small batches of a variety of products to be manufactured every day.

For small batches to be practical, the tooling has to be fast. That's why Taiichi Ohno said, "Our production slogan is small lot sizes and quick setups." Reducing setup times is important in all types of work so that you can focus on doing the actual work. Reducing setup times often adds flexibility that can change the way that the team works.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Another type of waste is not automating what should be automated. Toyota makes sure to automate where it makes sense. The same is true for any knowledge work tasks that can and should be automated.

Taiichi Ohno points this out by saying, "With computers available, it's a waste to perform calculations by hand."

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Building Quality In**

A very important Lean principle is to build quality into the product as you are building it. You don't want to start with poor quality and then have the mindset that the quality will improve over time.

As Taiichi Ohno says, "Produce quality products 100 percent of the time”.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto, Email

Descrição gerada automaticamente

It's important to start with a strong technical foundation of what you're doing and to never compromise on quality to save time.

Related to building quality in is the principle that the process should identify problems. You do not want problems to go unnoticed.

Taiichi Ohno describes this as distinctions between normal and abnormal operations must be clear and countermeasures or solutions always taken to prevent recurrence. This is one of the reasons for having the Kanban board at Toyota. If there's a problem, it is displayed for all to see and take appropriate action.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

We have also seen an example of the process identifying a problem by adding work in progress limits to the columns on the board.

When the team goes over the work in progress limit, the column turns red to signal the team that there is a problem and take appropriate action. Your team will have custom ways of identifying problems. It is important to fix problems when they are discovered. You don't want to build a list of problems to fix. This takes more time and discipline, but the result is much better quality.

Taiichi Ohno makes this point when he says, "Correct a mistake immediately, to rush and not to take time to correct a problem causes work loss later."

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

In his book, Taiichi Ohno describes the origin of the idea to fix problems when they are discovered.

Sakichi Toyoda was the founder of the Toyota Motor Company. Prior to that, he owned a company that weaved fabrics.

He invented an auto activated weaving machine. The machine would stop if any of the threads broke, an operator could then replace the thread.

Prior to this invention, if a thread broke, it was only discovered after the defective products were already made. In other words, the quality control was moved from after the product was built to during the building of the product.

Taiichi Ohno writes in his book, "Because a device that could distinguish between normal and abnormal conditions was built into the machine, defective products were not produced”.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

If you are familiar with the concept of test-driven software development, you will notice that this is the same principle. Any problems with the software are discovered and fixed as the software is being developed. When problem solving, it's important to identify and fix the root cause of the problem.

Taiichi Ohno talks about Toyota's five-whys approach to problem solving: By asking why five times and answering it each time, we can get to the real cause of the problem, which is often hidden behind more obvious solutions. So, the idea is to continue asking questions until you find the root cause of the problem and then fix the problem there.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

The number five is just a typical number of times the why question needs to be asked. You want to keep asking **why** until you aren't talking about a symptom but rather the root cause of the problem. Stopping early and fixing a symptom will usually mean that the root cause will create problems somewhere else.

Here is an example of asking why to find the root cause of a problem: Let's assume that our problem is that our product was delivered late:

* The first question that we could ask is, why was the product delivered late? The answer could be because we had too much to do at the end of the project. This is not a root cause of the problem because there could be many reasons why we had too much to do at the end of the project;
* The next question could be, why did we have too much to do at the end of the project? The answer could be because we started by researching and planning the entire project, and many of the plans changed after development started. This is getting closer to the root cause of the problem. It is identifying waste.

If you make detailed plans that end up changing and research topics that don't end up being part of the product, this is a form of waste. Sometimes things take longer because each task is unique, and we were optimistic in our prediction. Other times like this, sources of waste can be identified, and the process can be improved.

The next question could be, why didn't we plan and build incrementally? The answer could be that we probably should have. We have identified the root cause of the project delay. So, the solution in this case is to research, plan, build, and deliver incrementally.

An incremental approach to developing the product would not only have reduced waste, but also would allow us to be flexible with the scope of the project and still deliver on time.

That is an overview of some of the Lean principles. Taken together, they provide a way of working that is very effective in many contexts. If you look at the individual principles, you can see that they are timeless. The word Lean is currently used to describe them, but that word is not important and may not be universally used. We will see that Agile principles are very similar to Lean principles.

Lean Thinking is by the authors of a book called The Machine That Changed the World, which was many people's first introduction to the Toyota production system.

Lean Thinking discusses applying the thinking behind the Toyota production system more broadly.

Implementing Lean Software Development applies Lean ideas to software development.

The Lean Startup applies Lean ideas to the building and management of startups.

Lean Enterprise applies Lean ideas throughout the enterprise.

**Agile Manifesto**

O Agile Manifesto foi criado em 2001 e é um documento histórico estático.

Embora a melhoria contínua seja fundamental para a agilidade, o documento permanece estático devido ao acordo dos autores na época.

O manifesto representa a convergência de diferentes abordagens ágeis daquele período.

O escopo do manifesto é limitado ao desenvolvimento de software, mas muitas de suas ideias são aplicáveis a projetos ágeis em geral.

Value Statements do Agile Manifesto

O manifesto começa com uma explicação sobre a descoberta de melhores maneiras de desenvolver softwares e valorização de indivíduos e interações sobre processos e ferramentas.

É enfatizada a importância de empoderar a equipe e promover a colaboração com os clientes.

O manifesto também destaca a preferência por software funcionando em detrimento de documentação abrangente.

Além disso, valoriza-se a capacidade de responder a mudanças em vez de seguir rigidamente um plano.

**Princípios derivados dos Value Statements:**

Os value statements são transformados em princípios para uma melhor organização.

Os princípios derivados incluem: capacitar a equipe, abraçar a mudança, colaborar com o cliente e planejar, desenvolver e entregar incrementalmente.

**12 Princípios Ágeis:**

- Satisfazer o cliente por meio da entrega antecipada e contínua de software valioso;

- Acolher mudanças nos requisitos, mesmo tarde no desenvolvimento;

- Entregar software em funcionamento com frequência, de algumas semanas a alguns meses, com preferência para um prazo mais curto;

- As pessoas de negócios e os desenvolvedores devem trabalhar juntos diariamente durante todo o projeto;

- Construir projetos em torno de indivíduos motivados, fornecer o ambiente e o suporte de que precisam e confiar neles para fazer o trabalho;

- O método mais eficiente e eficaz de transmitir informações para uma equipe de desenvolvimento é a conversa face a face;

- O software em funcionamento é a medida primária de progresso;

- Os processos ágeis promovem o desenvolvimento sustentável;

- Contínua atenção à excelência técnica e bom design aprimora a agilidade;

- A simplicidade, ou seja, a arte de maximizar a quantidade de trabalho não realizado, é essencial;

- As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizadas;

- Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e ajusta seu comportamento de acordo.

**Obs.:** A reorganização dos princípios visa torná-los mais aplicáveis em contextos mais amplos, além do desenvolvimento de software. As equipes e organizações podem adaptar esses princípios conforme necessário.busca pela simplicidade e auto-organização da equipe, além de reflexão e adaptação contínuas.

**Lean versus Agile: Explaining the Principles**

Introduction to Lean and Agile:

Lean is a term coined by John Krafcik at MIT and refers to the ideas of the Toyota Production System.

Lean principles can be applied to any type of project, even personal endeavors.

Agile, on the other hand, was introduced by the participants who created the Agile Manifesto in 2001.

Initially, Agile was used to describe a lightweight alternative to plan-driven or Waterfall Software Development.

However, over time, Agile has evolved into a broader concept applicable to various project domains.

While Lean could have been called Agile and vice versa, both terms are often used interchangeably to describe an adaptive approach that prioritizes value.

**Common Principles between Lean and Agile:**

Many principles found in Taiichi Ohno's book "Toyota Production System" align with Lean and Agile methodologies.

Here is a combined list of principles that are shared between Lean and Agile:

a. Empowering the team: Both Lean and Agile emphasize empowering the team to make decisions and take ownership of their work.

b. Visualizing work: Visual management techniques, such as Kanban boards, are used in Lean and Agile to provide transparency and promote collaboration.

c. Experimenting with the scientific method: Both Lean and Agile encourage a culture of experimentation, learning from failures, and applying the scientific method to drive continuous improvement.

d. Planning, developing, and delivering incrementally: Both methodologies advocate for iterative and incremental approaches to project development, allowing for flexibility and adapting to changes.

e. Improving the flow of value: Lean and Agile focus on optimizing the flow of value through the project by eliminating waste, reducing bottlenecks, and enhancing efficiency.

f. Building quality in: Quality is a key aspect in both Lean and Agile, with an emphasis on early and continuous testing, feedback loops, and a proactive approach to preventing defects.

**Importance of Understanding and Applying the Principles:**

Understanding and applying these principles are crucial for successful implementation of Lean and Agile methodologies.

By embracing these principles, teams can create a collaborative and empowered environment that promotes efficiency, flexibility, and value delivery.

However, it's important to apply the principles where appropriate, considering the specific needs and context of the project at hand.

By following these principles, organizations and teams can effectively leverage Lean and Agile methodologies to achieve their goals and continuously improve their processes.

**Lean e Agile:**

Lean é um termo cunhado por John Krafcik no MIT para descrever o sistema de produção da Toyota.

O Lean pode ser aplicado a qualquer tipo de projeto, inclusive pessoal.

O termo Agile foi criado pelos participantes que desenvolveram o Agile Manifesto em 2001.

Inicialmente, o Agile descrevia uma alternativa leve ao desenvolvimento de software baseado em planejamento.

Com o tempo, Agile ganhou um significado mais amplo, podendo ser aplicado a qualquer tipo de projeto.

Tanto Lean quanto Agile têm princípios em comum e podem ser usados de forma intercambiável para descrever uma abordagem adaptativa que foca em valor.

**Princípios combinados de Lean e Agile**

Os princípios de Lean e Agile são apresentados em uma lista combinada.

Esses princípios incluem:

- Capacitar a equipe;

- Visualizar o trabalho;

- Experimentar usando o método científico;

- Planejar, desenvolver e entregar incrementalmente;

- Melhorar o fluxo de valor;

- Promover a qualidade.

**Essential search methods in Jira:**

**Quick search and basic search**. We'll explore the various ways you can search in Jira, including project boards, reports, and dashboards.

**Quick search** is a versatile feature that allows you to search for issues, boards, projects, and more. It provides instant results as you type and offers options to exclude terms and use logical operators.

On the other hand, **basic search** utilizes user interface elements for searching issues. It includes features like limiting searches to specific projects, searching by field values, and performing date-based searches.

Additionally, Jira offers filters and quick filters for convenient saved searches and limiting the visibility of issues on boards.

Searching is crucial as it allows you to customize your Jira experience to match your team's desired process. A powerful search functionality often operates behind the scenes, supporting specific project views.

**Quick search**:

It is a global search available in both the cloud and server versions of Jira. When you click on quick search, you will initially see recent items. As you type, the results will dynamically update.

As you begin typing in quick search, you will see immediate results. For example, if you search for "item," Jira will display three issues with the word "item" in their text-based fields.

To refine your quick search, you can use keywords like "NOT" to exclude certain terms or "OR" to search for issues containing specific keywords. For instance, you can search for issues with either the word "item" or "sample" in their text-based fields.

**Basic search**:

It utilizes a row of user-friendly interface elements. When searching for all issues, none of the elements are selective.

You can limit your search by specifying the project, such as searching for issues in Project A only.

Basic search also includes a text box where you can search for specific values within issue fields. If the desired field is not shown, you can use the "more" dropdown menu to search by other fields, like priority.

For example, you can easily search for issues updated within the last three days using basic search. Date-based searches are straightforward with this feature.

To recap, quick search can search the text of issues, board names, project names, and filter names. Basic search offers a user-friendly way to search for issues using various interface elements.

**JQL – Jira Query Language:**

Let's start with an overview of **JQL (Jira Query Language)**. We want to search for all the issues in project A that we have permission to view. In basic search, there is a "**Switch to JQL**" link that allows you to switch to the advanced mode of searching. In advanced search or JQL search, Jira automatically populates the textbox with a text-based equivalent of the current basic search. You can click the "**Switch to basic**" link to switch back to basic search.

The text we see in the advanced issue search is called Jira Query Language or **JQL**, which uses a structured **query language-like syntax**. JQL is used to search issues only, unlike SQL (Structured Query Language) which is used to search databases.

Basic search provides a user-friendly interface for performing searches, while advanced search using JQL allows for more powerful searches beyond the capabilities of the basic search interface. If you are automating tasks related to Jira, JQL is the preferred choice because automation relies on text-based scripts. Whether you use basic or advanced search, JQL is always present behind the scenes.

Writing JQL queries doesn't always require manual input. Jira can generate the queries for you.

In basic search, you can select a project from the dropdown menu, and then click on the "Advanced" link to enter advanced search. The search results remain the same, but now the query is displayed instead of the basic search user interface. The displayed query shows all the issues with a project key of PROJ, sorted by the creation date. JQL queries can perform two fundamental actions: selecting a subset of issues based on specific criteria and sorting the results using an "ORDER BY" clause.

**Autocomplete**:

When creating a JQL query from scratch, you can start typing the field name in the Advanced Search textbox and Jira will provide autocomplete suggestions. Autocomplete shows up to 15 matches, making it easier to write JQL and reduce errors.

For example, if we type "pr," Jira suggests the "project" field. By pressing the spacebar, Jira displays a list of operators to choose from. We can select the "equals" operator to specify the project we want to search. Jira also shows a list of available project names based on our permissions. After completing the query and pressing "Enter," the search is performed. The basic search dropdown will reflect the project selection made in the JQL query.

To modify the order of search results, we can use the "ORDER BY" clause in JQL. An empty JQL query, like an empty string, returns all the issues and uses the default ordering, which is typically by issue key in descending order. By adding an "ORDER BY" clause to the query, we can specify the sorting order based on a particular field.

Clicking on column headers in the search results changes the "ORDER BY" clause accordingly. Multiple field names can be added to the "ORDER BY" clause separated by commas. The first field is the primary sorting field, and subsequent fields are used if the primary field has multiple values that are the same.

**JQL Functions**:

Functions are small programs that Jira executes before running the query. Their results are substituted into the query. Using functions allows for dynamic query creation rather than hardcoding information.

For example, a function can be used to search for issues assigned to the currently logged-in Jira user.

The Jira documentation provides details on the available advanced search functions, and you can find them by searching for "advanced searching functions reference" on Atlassian's website.

When using JQL (Jira Query Language), you can find details about the functions that interest you. Here are some **advanced search functions related to time and date**:

* Functions for specifying the start and end of a time period;
* The "now" function represents the current time;
* The "current login" function gives the time when the current user logged into the session;
* The "last login" function provides the time when the current user logged into the previous session.

For example, to find issues created since the start of today, you can use a query that includes the current date. Jira allows you to use time unit qualifiers to specify relative dates in the query. These qualifiers are special strings that Jira replaces with actual date values before running the query.

You can search for issues created in the last two days by using the time unit qualifier "2d" in the query. The "d" represents days. Similarly, you can search for issues created in the past two weeks by using "2w" as the qualifier.

Jira provides autocomplete suggestions for operators based on the field you're working with. The equals sign (=) can be used for the "project" field, while the tilde (~) represents the "contains" operator for the "summary" field. Autocomplete helps you choose the appropriate operator for each field.

Operators such as equals, not equals, greater than, greater than or equal, less than, less than or equal, in, not in, is, is not, contains, does not contain, was, was not, was in, was not in, and changed are commonly used in JQL queries. The operators can be used to compare field values and perform specific search operations.

**Boolean operators** are essential for combining or negating clauses in your JQL queries. The two primary Boolean operators are AND and OR, which allow you to refine your search by combining multiple clauses. For instance, you can find all issues assigned to the current user with the status of "In Progress" or find issues with the status of "Selected for Development" or "In Progress." The latter query is equivalent to using the IN operator.

On the other hand, the NOT operator is used to negate clauses. For example, you can search for issues that do not have a status of "Backlog." Additionally, you can combine AND and NOT Boolean operators, or you can use OR and NOT Boolean operators to achieve similar results. If your query includes multiple Boolean operators, it's advisable to use parentheses to establish operator precedence and enhance query readability.

Consider this example: by adding parentheses, the OR Boolean operator is evaluated first, and the result is then combined with the first clause. This query retrieves all issues with the status of "Selected for Development" or "In Progress" and the summary containing the text "item," resulting in eight issues. However, removing the parentheses causes the AND Boolean operator to be evaluated first. Consequently, the query searches for issues containing the text "item" with the status of "Selected for Development" or any issue with the status of "In Progress." In this case, one additional issue is included, even if the word "item" is not present in the summary.

When combining AND and OR Boolean operators in a query, there are two precedence rules to keep in mind. First, parentheses take precedence, ensuring the evaluation of what's inside them first. Second, the AND operator normally takes precedence over OR, meaning that if no parentheses are used to modify the precedence, the AND operators are evaluated first from left to right. This behavior is akin to mathematical operations, where multiplication has precedence over addition unless parentheses are used to alter the precedence.

Here's an example: suppose your query involves both OR and AND operators without parentheses to modify the precedence. In this scenario, the AND operator will be evaluated first because it has precedence over OR. This query is equivalent to the previous one. However, by enclosing the AND operator in parentheses, you explicitly instruct the query to evaluate the AND operator first, although it was already being evaluated first without parentheses. As a result, the two queries are equivalent. Remember that using parentheses in queries with mixed operators can help maintain simplicity and remove any ambiguity.

To summarize our discussion, Boolean operators play a crucial role in combining or negating clauses in JQL queries. By leveraging AND and OR operators, you can refine your searches effectively. Additionally, using the NOT operator allows you to exclude specific criteria. Parentheses can be used to set operator precedence and improve query readability. Remember to consider the two precedence rules: parentheses take precedence, and the AND operator typically has precedence over OR. By following these guidelines, you can create powerful and precise JQL queries for your needs.

**Filters: Creating, Describing Board Filters, and Using Quick Filters:**

When observing the interface, you'll notice a list of tabs located on the left side, referred to as filters. These filters serve as saved searches that enable you to swiftly execute specific search queries. In the demonstration, the "my open issues" filter is selected as an example. However, you have the freedom to create your own filter by initiating a search and then saving it by clicking on the "Save as" option.

After initiating the save process, a prompt will appear, requesting a name for the filter. In this particular case, the filter is assigned the name "My in progress".

Once the filter is created, it will be displayed in the sidebar, specifically under the "starred" category. By clicking on the "Details" link situated to the right of the filter name, you can access and modify the specific details associated with the filter.

Within the filter details section, you'll find an "Edit permissions" link. Clicking on this link will open a dialogue box in the upper right corner, providing you with the ability to specify who has access to the filter. In the current example, the filter is set to remain private.

Additionally, clicking on the "New subscription" link will reveal another dialogue box in the lower right corner. Subscriptions allow recipients to receive query results via email at a frequency determined by the subscription settings. The recipients can be individual users or members of Jira groups. In this particular case, the subscription is personal, and the query results will be emailed once per day.

Furthermore, you can access all filters in the sidebar by clicking on the "View all filters" option. By clicking the "more" icon located to the right of a filter, you can modify the metadata associated with that filter.

At this point, you can select a filter and modify the search criteria as desired. Once the necessary changes are made, you will observe that the filter has been edited, and you can save the changes by clicking on the "Save" button.

**Board Filters:**

Every board within Jira incorporates a filter that determines which issues are displayed on that specific board. In the board settings, you have the ability to edit the board's filter, allowing you to customize the issues presented on the board. When creating a board, it is necessary to assign it to either a new or an existing filter. You can see the query utilized in the background for a board. Specifically, it selects all issues with the project key "PRJCT" and orders them based on their rank.

Board filters can be employed to showcase issues from multiple projects on a single board. On the left side of the screen, under the "Filter Query" section, you can observe that the JQL includes issues from projects with the key "PRJCT" or "PRJ". As displayed on the right side of the screen, when viewing the board, you will notice the presence of issues originating from multiple projects.

**Quick Filters:**

Quick filters provide a means to further refine the issues displayed on a board. By default, all issues that match the board filter are shown. In the video, six quick filters are demonstrated. Clicking on a user's icon will display only the issues assigned to that particular user.

Additionally, you can utilize quick filters such as "Only My Issues" or "Recently Updated" to narrow down the displayed issues. It's worth noting that the board also incorporates a search box, allowing you to conduct a text-based search to further limit the issues shown.

Quick filters can also be utilized to refine reports. In the example provided, a cumulative flow diagram is showcased, and by selecting the "Refine Report" drop-down option, you can observe the ability to select a quick filter to further fine-tune the report. In this specific case, the report will display older issues that are not yet complete.

To summarize, filters are saved searches that can be accessed through user interface elements, and every board incorporates a filter that determines the displayed issues. Quick filters, on the other hand, are saved searches utilized to further limit the issues shown on a board or in reports.