WHITE PAPER

Understanding Agile Project Definition

Clarifying terms with stakeholders and adjusting development methods to aid communication helps achieve the best outcomes from Agile projects.

RG Robbins-Gioia, LLC

TEAMRG.COM

Key Points

- Epics, User Stories, and Use Cases differ in their level of detail and purpose for understanding project requirements and developing a solution.
- Utilize the 6 P's: Proper Planning and Preparation Prevents Poor Performance when breaking down your project into concepts like Epics, User Stories, Use Cases.
- Iterative definition aids in teamwork and collaboration while remaining agile and adjustable to changing requirements, markets, or business objectives.

To achieve their strategic goals and keep up with a rapidly changing operational, economic, and technological landscape, many organizations have embraced Agile methodologies as part of modernizing their enterprise systems and streamlining complex business processes. However, to be successful this requires all participants in the Agile process understand the terminology and techniques being applied to deliver an Agile modernization project.

Critical to the success of a project is establishing a common understanding of the organization's needs, workflows, and desired outcomes. The project team and its stakeholders must agree on how to gather requirements, see the system from multiple users' points of view, and resolve existing process problems in order to build a system which is easy to use and delivers the expected business benefits.

Agile applies different techniques for project definition and requirements capture than a traditional waterfall method. Rather than creating detailed requirement documents and systems specifications, Agile introduces the concept of persona-centric definition of workflows and features through epics, user stories, and use cases. These constructs are at the core of successful Agile implementation, so it is essential they are well understood within the team and by stakeholders, including project sponsor, managers, and end users.

For a team new to Agile – or some who may have been practicing Agile poorly –

these terms can create misunderstandings with project stakeholders, who may not fully understand what they mean and how they are applied. Failing to establish a common understanding can result in various contributors having different views of what the terms mean or opinions on when and where these concepts should be applied, potentially resulting in poor definition, unclear workflows, and a disorganized backlog.

For more guidance on how to successfully run an Agile project, visit this Agile Practitioners Guide.

This will likely result in constant review and rework both of requirements and the system capability being implemented. This additional rework can act as a drag on "velocity", one of the key measures of Agile project productivity, and undermine the likelihood of successful delivery on time, to budget, and with the required capabilities.

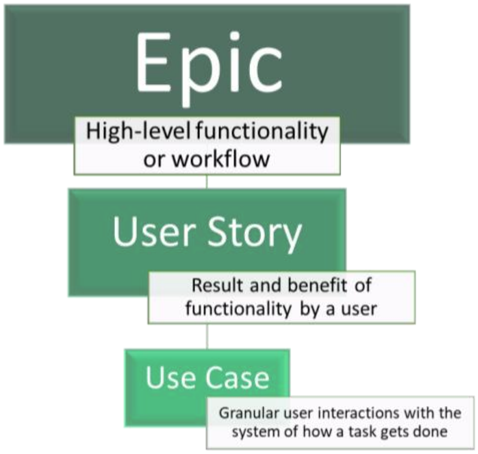
To avoid this, it is important to establish a consistent working knowledge of the differences between epics, user stories, and use cases across the entire project team. While they form a coherent hierarchy which describes the system to be implemented, each differ in their level of complexity and relevance to the project.

A common understanding of the concepts is important for facilitating collaboration, increasing productivity, and maintaining a robust understanding of user needs traceable to the overarching organizational objectives.

Which One is Which?

The basic Agile systems description forms a set of collections in increasing levels of detail: *The **System** is composed of **Epics** which are made up of **User Stories** that are broken into **Use Cases***.

Rather than try and describe the entire system all at once with details requirements and specifications - as tended to be the approach in traditional Waterfall projects – in Agile the principle is to describe the system broadly, and then refine the highest priority areas first, steadily working with the end user communities to validate needs in close parallel with actual development. This means that at any point in time some parts of the system may be described in detail, while others will only be at a high level.


Epic
High-level functionality or workflow
User Story
Result and benefit of functionality by a user
Use Case
Granular user interactions with the system of how a task gets done

In practice, for enterprise systems projects some fundamentals need to be done first to create a secure, stable, scalable platform. This includes the systems, integration, data, and security architectures which ensure the ability for the system to connect to other business systems and protect the information held within it. Failing to do this well frequently burdens an Agile project by implementing a user interface on top of a flawed architecture which then become expensive in both time and cost to fix.

An Epic describes a broad function or workflow of the system.

In software development an Epic is not a heroic story in poem form but a short description of a large scale or broad function in the system. In most projects, it is the first step to organizing what the system will do in terms of a coherent set of related functional blocks.

Like all Agile descriptions, an Epic is written from the perspective of the *user personas*. The collection of Epics should cover all the functional and non-functional requirements of the system and be readable from the perspective of the business user.

There are various ways Epics can be created and shared, ranging from purpose-built tools such as Jira to shared documents and Wikis and physical representations such as colored notecards stuck to a wall in the development team's room, often next to the Kanban board. This allows the entire structure of the system description to be treated as a "living repository" which is constantly updated and reviewed.

This article mentions how Epics are an ideal way for communicating with non-technical stakeholders to gain their feedback and contribution, and thus help the team understand the large scale functions the system.

As the Epic is broadest baseline for an Agile system description, it is not usually completed within one sprint and often multiple people will contribute to building it. It is useful for tracking progress, as it inherits the estimates from its User Stories and Use Cases, and so provides a summary of the time, personnel, and budget needed to complete the project.

Human and Non-human Personas. One technique for capturing both functional and non-functional requirements is to create "non-human" personas to capture the full breadth of capabilities required of the system but outside or abstracted from any particular "human" persona. For example, if the system interacts with another business system, for example the financial management system, the FMS can be treated like a persona and described in terms of its interactions, workflows, and results.

A User Story describes the experience of a specific Persona in completing an activity

As is clear from its title, a user story focuses on the impact and interaction of different *user personas* with the system. They identify the role of the user and why this interaction is important to that user. They are often written something like this, "as a *[user persona]* I want to *[do activity]* because *{desired outcome]*."

In many enterprise-level projects, User Stories are captured by Business Analysts using elicitation techniques with stakeholders to help identify, verify, and confirm requirements. The Business Analysts are often supplemented by Customer Experience (CX) and User Experience (UX) designers who start interpreting and visualizing how the system will appear and be adopted by the end users through journeys, mock-ups, and interactive prototypes. User Stories help the development team understand in detail the workflows and needs end user users have for certain functions.

User Stories are more detailed than Epics as they include more granular descriptions of user workflows to describe how and why a user will use system. A key aspect is while they describe the *activity* and *benefit* of what the user needs to accomplish, they do not describe the *actions* the user takes – that is the purpose of the Use Case.

This article from *Medium* discusses the purpose of User Stories in more detail and includes an evaluation method called INVEST.

A Use Case sets out how a user will complete a specific task.

A Use Case is a detailed description of how one user interacts with one function of the system. Use cases are behavior-focused, identifying the steps the user takes to accomplish one task. Because they are more detailed, they are often completed with models showing the normal flow of behaviors a user would do. The detail can be very useful, but they take more time to create and any adjustments or changes to the system can take a long time to complete. There can be a thorough documentation process of these behaviors for each function.

For this reason, Use Cases are often developed close to the time the development team will start implementing them. Where possible, the CX/UX designers and user interface (UI) designers will often work to define a related set of User Stories and their Use Cases in the project backlog, even if not all will initially be implemented. This helps the designers most efficiently work on blocks of related functionality and sets the team up for higher velocity in the future.

For more information on pros and cons of Use Cases visit this article.

Why Does it Matter?

Having the team and stakeholders understand why Agile uses Epics, User Stories, and Use Cases to describe the system and deconstruct user requirements is an essential part of explaining other Agile concepts such as the backlog, Kanban, sprint, burndown, and velocity.

This blog includes broad definitions for many terms commonly used in Agile.

For business stakeholders unfamiliar with Agile, it can often feel risky or chaotic to start development without a complete understanding of the system specifications and requirements and their attendant detailed estimates for time, effort, and cost. The initial planning and definition work to create the Epics, define the architecture, and set the development standards helps building confidence in the Agile process and establishing the framework for a successful Agile project.

The 6 P's: Proper Planning and Preparation Prevents Poor Performance

Spending time up front to ensure that everyone, from stakeholders to team members, understands the objectives and methodology is an Agile best practice. Stakeholders and team members may think they know Agile methods and terminology but in fact may have fallen victim to "Agile BS." Explaining and clarifying methods can help establish the relative levels of knowledge across the team and build a constructive relationship based around common principles, and avoid the temptation to "be agile" by rushing into coding poorly described products.

Visit this article for how to detect "Agile BS."

The traditional time-consuming process of gathering and understanding requirements is made more efficient by applying Epics, User Stories, and Use Cases and delivering material for review and feedback on an iterative basis. Using "maps" of the system definition helps stakeholders and the project team understand priorities, dependencies, and progress by providing an evolving visualization of the current effort and backlog. These graphical representations of interactions, benefits, and functions of systems and why they are important to users provides clarity of the solution scope.

Understanding the different uses that each term has in development can assist the team in adjusting to last minute functions and changing requirements. It is also useful in understanding how much time and work are associated with those changes. In total, Epics, User Stories, and Use Cases create a comprehensive description of the system solution that displays the team's understanding of stakeholder needs and business requirements.

Visit here for tips on writing user stories and use cases that fit your project and objective.

Iterative Definition Aids Collaboration and Teamwork

The Business Analyst's role is to gather, verify, and understand requirements and communicating them with the rest of the development team. In an Agile project, it may be a team effort to meet with stakeholders and discuss the solution. Creating an organization of the high-level functional requirements to the interactions users have with the system functions help all team members be on the same page with the purpose and features of the solution.

Teamwork can be fostered with the use of Epics, User Stories, and Use Cases by helping to break down activities within the team and creating a timeline for each workstream, leading to better budgeting. The range from broad to specific action items create tasks that help keep the backlog organized and avoid rework. Understanding what these terms are and how they relate to each other and to the final project is essential for new members to the development team. Unexperienced personnel get a visual representation of functions and can understand the different levels of detail as well as the importance of each feature. Epics, User Stories, and Use Cases help organize activities and help team members be clearer with what coding needs to be done.

This article discusses how standups and sprint planning can help validate and clarify functionalities and requirements with stakeholders as team members elaborate on User Stories with Use Cases.

Agile Definition Adapts to Projects and Progress

Epics, User Stories, and Use Cases vary in their level of complexity and formality, allowing the team to adjust their practices depending on the project's scope and needs. Different projects will require different acceptance criteria, documentation requirements, and personnel, and may not need to use all three levels.

Understanding how to apply the concepts of Agile definition will allow the team to assess the functions needed in the project and create an ordered path to the best solution that displays the features of the system. There is no exact formula for a perfect Agile project, by nature – it is important the team and methods be able to adjust to project needs.

In Sum

Improving enterprise business systems in today's operational environment can be complex and time consuming, placing great responsibility on project teams to deliver the benefits from modernization projects rapidly and effectively.

While Agile is a well proven approach for implementing business systems, in the rush to "deliver working software" project teams and stakeholders may not understand how to apply the Agile concepts of Epic, User Story, and Use Cases to effectively capture and communicate the requirements, validate them with the business, and continuously refine and track them against priorities.

Having an understanding of how these concepts are applied helps improve teamwork and collaboration and allow the project team to progress through the development process while allowing for flexibility and innovation, resulting in the most efficient and encompassing solution.

Interested in learning more about Agile Program Management? Visit teamrg.com and visit our resource center or contact us.