The iron triangle of planning

The ultimate balancing act and how to achieve agile project management nirvana

BY TAREQ ALJABER

BROWSE TOPICS

Agile manifesto
Scrum
Kanban
Agile project management
Product Management
Agile at scale
Overview
Managing an agile portfolio
Lean portfolio management
Okrs
Using Improvement Kata to support lean
The Large-Scale Scrum (LeSS) framework
Beyond the basics whitepaper
Software development
Design
The agile advantage
DevOps
Agile Teams
Agile tutorials
About the Agile Coach
All articles

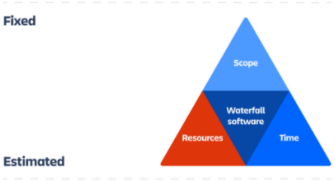
All agile software projects have goals: what the project needs to deliver, when it needs to be delivered by and within what budget. However, managing these three constraints can be a complex juggling act. So let's take a cue from the decades-old iron triangle of planning and learn how balancing different variables can help agile software teams achieve agile project management nirvana.

The traditional iron triangle

The iron triangle models constraints of project management and these constraints are considered "iron" because you can't change one constraint without impacting the others. The original iron triangle, proposed by Dr. Martin Barnes in 1969, follows a waterfall approach to product development: scope is fixed and resources and time are variable. For a software team, this would mean that teams start a project by defining product requirements to determine a project's scope (a list of work items). The resources and schedule are variable and are estimated depending on the fixed scope.

> CONSTRAINTS OF THE IRON TRIANGLE
>
> - Scope is the work to be done – such as features and functionalities – to deliver a working product.
> - Resources include budget and team members working to deliver and execute.
> - Time is when teams will deliver to the market such as releases and milestones.

The purpose of the iron triangle is to give product teams the necessary information to make trade-offs that will help the business. For example, if teams are faced with a fixed scope, they might be halfway through a project and realize that they won't hit their release date. The only variables they can play with are: 1) Time - they can accept a later release date or 2) Resources - they can add some more people to the project, which will increase costs. As software development evolved in the 21st century, the need for better collaboration and the ability to respond quickly to customer feedback became crucial, and thus, the agile methodology was born.



> Scope, people, and time are the basic building blocks of any plan.

Mapping the iron triangle to agile

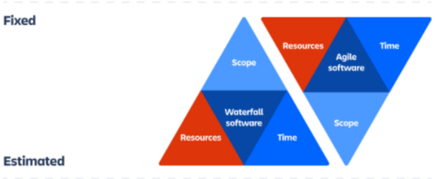
If you're a team that practices waterfall development or new to agile development, the important thing to remember is the difference between what is fixed and what is estimated. Unlike waterfall development, agile projects have a fixed schedule and resources while the scope varies. While the scope of a project might change in agile development, teams commit to fixed iterations of work: sprints if you're using a scrum framework, WIP limits if you're using a kanban framework. It's also a best practice to keep teams fixed throughout the development process. By keeping teams consistent on a product or project, they become more efficient through developed trust and

RELATED READING

Long-term agile planning - 8 steps to get started

Learn more →

continuity.



The idea of scope is the same in agile development: what software to build and deliver. However, agile focuses on high-level requirements rather than trying to come with deep and detailed requirements upfront. The scope of a project gets regularly managed and groomed (prioritized) by the product manager in a tool like Jira Software. The product manager decides which work should be accomplished in the next sprint based on agile qualitative and quantitative feedback from various channels (market conditions, customer feedback, competitions, etc.). And because resources and time are fixed, it's easier for development teams to react to market changes and to deliver value to customers faster. This transparency of constraints keeps teams honest about a consistent and fast release cadence, which is a key tenant of agile development; and by looking at projects through the lens of the iron triangle teams are able to adapt without abandoning a plan.

> > Set up your first agile project with this interactive tutorial

Long-term agile planning and the iron triangle

As projects become bigger, more teams are needed and the time box gets longer. Thus, the notion of fixing resources and time, while scope varies, is not a valid approach for all agile projects. Long-term agile planning requires a more flexible iron triangle that allows teams to plan ahead and ensures that they're meeting the business objectives. Think for instance about the lean startup movement, and the notion of a minimum viable product (MVP). An MVP by definition is a small set of features (scope) that delivers customer value. To get to that MVP, teams might need to stick to a fixed scope – the number of features – with time being their only variable (e.g. you can't release without certain features, so the release date gets pushed). Only after launching the MVP, teams switch to a variable scope.

Regardless of the differences between waterfall and agile development, when using the iron triangle, there's no right or wrong way. It's there to help you make the best decisions and trade-offs to reach your business goals. A tool like Portfolio for Jira visualizes the building blocks of a plan – scope, people, and time – to help teams plan in real-time. You can easily play with scope, teams and time to plan your next product release, using the team's existing data in Jira Software.

SHARE THIS ARTICLE

BROWSE TOPICS

Agile manifesto
Scrum
Kanban
Agile project management
Product Management
Agile at scale
Overview
Managing an agile portfolio
Lean portfolio management
Okrs
Long-term agile planning
What is SAFe ?
Organizational agility with

TUTORIAL

Learn scrum with Jira Software

A step-by-step guide on how to drive a scrum project, prioritize and organize your backlog into sprints, run the scrum ceremonies and more, all in Jira.

Try this tutorial →

ARTICLE

Agile and DevOps - Friends or Foes?

DevOps is agile applied beyond the software team. Read our guide on how agile, DevOps and continuous delivery all work together (or not).

Read this article →

Agile Topics

Agile project management
Scrum
Kanban
Design

Software development
Product management
Teams
Agile at scale

Sign up for more agile articles and tutorials.

Email

DevOps

email@example.com

Subscribe

Up Next
Using Improvement Kata to support lean →