**Scrum**

**Scrum** is a framework for developing, delivering, and sustaining products in a complex environment,with an initial emphasis on [software development](https://en.wikipedia.org/wiki/Software_development), although it has been used in other fields including research, sales, marketing and [advanced technologies](https://en.wikipedia.org/wiki/High_tech). It is designed for teams of ten or fewer members, who break their work into goals that can be completed within time-boxed iterations, called *sprints*, no longer than one month and most commonly two weeks. The Scrum Team assess progress in [time-boxed](https://en.wikipedia.org/wiki/Timeboxing) daily meetings of 15 minutes or less, called daily scrums (a form of [stand-up meeting](https://en.wikipedia.org/wiki/Stand-up_meeting)). At the end of the sprint, the team holds two further meetings: the sprint review which demonstrates the work done to [stakeholders](https://en.wikipedia.org/wiki/Stakeholder_(corporate)) to elicit feedback, and [sprint retrospective](https://en.wikipedia.org/wiki/Retrospective#Software_development) which enables the team to reflect and improve.

Scrum is a lightweight, [iterative](https://en.wikipedia.org/wiki/Iterative_design) and [incremental](https://en.wikipedia.org/wiki/Iterative_and_incremental_development) framework for developing, delivering, and sustaining complex products. The framework challenges assumptions of the traditional, sequential approach to product development, and enables teams to self-organize by encouraging physical [co-location](https://en.wikipedia.org/wiki/Colocation_(business)) or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines involved.

A key principle of scrum is the dual recognition that customers will change the scope of what is wanted (often called *requirements volatility*) and that there will be unpredictable challenges for which a predictive or planned approach is not suited. These changes come from a variety of sources, but according to scrum, understanding why is irrelevant, and change should simply be accepted, embraced and analyzed for benefits.

As such, scrum adopts an evidence-based [empirical approach](https://en.wikipedia.org/wiki/Empirical_process) accepting that the problem cannot be fully understood or defined up front, and instead focusing on how to maximize the team's ability to deliver quickly, to respond to emerging requirements, and to adapt to evolving technologies and changes in market conditions.

**Benefit of Scrum in Software Development**

**Better quality**

Projects exist to accomplish a vision or goal. Scrum provides the framework for continual feedback and exposure to make sure that quality is as high as possible. Scrum helps ensure quality by the following practices:

* Defining and elaborating on requirements just in time so that knowledge of product features is as relevant as possible
* Incorporating daily testing and product owner feedback into the development process, allowing the development team to address issues while they’re fresh
* Regular and continuous improvement of scrum team output (product or service) through sprint reviews with stakeholders
* Conducting sprint retrospectives, allowing the scrum team to continuously improve such team-specific factors as processes, tools, relationships, and work environments
* Completing work using the definition of done that addresses development, testing, integration, and documentation.

**Decreased time to market**

Scrum has been proven to deliver value to the end customer 30 to 40 percent faster than traditional methods. This decrease in time is due to the following factors:

* Earlier initiation of development due to the fact the upfront documentation phases of waterfall projects (which typically take months) are foregone by having a dedicated product owner embedded within the scrum team to progressively elaborate requirements “just in time” and provide real-time clarification.
* Highest-priority requirements are separated from lower-priority items. Incrementally delivering value to the end customer means that the ­highest-value and -risk requirements can be delivered before the lower-value and risk requirements. No need to wait until the entire project is complete before releasing anything into the market.
* Functionality is swarmed to completion each sprint. At the end of every sprint, scrum teams produce working product and service increments that are shippable.

**Increased return on investment**

The decrease in time to market is one key reason that scrum projects realize a higher return on investment (ROI). Because revenue and other targeted benefits start coming in sooner, earlier accumulation means higher total return over time. This is a basic tenet of net present value (NPV) calculations. In addition to time-to-market benefits, ROI with scrum also increases by

* Regular feedback through sprint reviews directly from stakeholders, including customers, enables course corrections early, which is less costly and time-consuming than later in the process.
* Fewer costly defects due to automation and up-front testing means less wasted work and faster deployments.
* Reducing costs of failure. If a scrum project is going to fail, it fails earlier and faster than waterfall projects.

**Higher customer satisfaction**

Scrum teams are committed to producing products and services that satisfy customers. Scrum enables happier project sponsors through the following:

* Collaborating with customers as partners and keeping customers involved and engaged throughout projects.
* Having a product owner who is an expert on product requirements and customer needs.
* Keeping the product backlog updated and prioritized to respond quickly to change.
* Demonstrating working functionality to internal stakeholders and customers in every sprint review.
* Delivering product to end customers faster and more often with every release rather than all at once at the very end.
* Incrementally funding projects instead of requiring large up-front commitments.

**Higher team morale**

Working with happy people who enjoy their jobs can be satisfying and rewarding. Self-management puts decisions that would normally be made by a manager or the organization into scrum team members’ hands. Scrum improves the morale of team members in these ways:

* Being part of a self-managing and self-organizing team allows people to be creative, innovative, and acknowledged for their expertise.
* Development teams may organize their team structure around people with specific work styles and personalities.
* Scrum teams can make decisions tailored to provide balance between team members’ professional and personal lives.
* Having a peer relationship with a business representative (product owner) on the same team aligns technical and business priorities and breaks down organizational barriers.
* Having a scrum master, who serves the scrum team, removes impediments and shields the development team from external interferences.
* Focusing on sustainable work practices and cadence ensures that people don’t burn out from stress or overwork.
* Working cross-functionally allows development team members to learn new skills and to grow by teaching others.
* Encouraging a servant-leader approach assists scrum teams in self-management and actively avoiding command-and-control methods.
* Providing an environment of support and trust increases people’s overall motivation and morale.
* Having face-to-face conversations helps reduce the frustration of miscommunication.
* Ultimately, scrum teams can agree on rules about how they work to get the job done.

**Increased collaboration and ownership**

When scrum teams take responsibility for projects and products, they can produce great results. Scrum teams collaborate and take ownership of quality and project performance through the following practices:

* Having the development team, the product owner, and the scrum master work closely together on a daily basis
* Conducting sprint planning meetings, allowing the development team to organize its work around informed business priorities
* Having daily scrum meetings where development team members organize around work completed, future work, and roadblocks
* Conducting sprint reviews, where the product owner outlines his prioritization decisions and the development team can demonstrate and discuss the product directly with stakeholders
* Conducting sprint retrospectives, allowing scrum team members to review past work and recommend better practices with every sprint
* Working in a colocated environment, allowing for instant communication and collaboration among development team members, the product owner, and the scrum master
* Making decisions by consensus

**More relevant metrics**

The metrics that scrum teams use to estimate time and cost, measure project performance, and make project decisions are often more relevant and more accurate than metrics on traditional projects. On scrum projects, metrics are more relevant because

* Those who will be doing the work, and no one else, provide effort estimates for project requirements.
* Timelines and budgets are based on each development team’s actual performance and capabilities.
* Using relative estimates, rather than hours or days, tailors estimated effort to an individual development team’s knowledge and capabilities.
* In less than one minute a day, developers can update the burn-down chart, providing daily visibility of how the development team is progressing toward a sprint goal.
* At the end of every sprint, a product owner can compare the project’s actual cost (AC) plus the opportunity cost of future projects (OC) against the value that the current project is returning (V) to know when to terminate a project and begin a new one. You don’t need to wait until the end of a project to know what its value is.

**Improved progress visibility and exposure**

On scrum projects, every member of the project team (which includes the scrum team and stakeholders) has the opportunity to know how the project is going at any given time. Transparency and visibility make scrum an exposure model to help the project team accurately identify issues and more accurately predict how things will go as the project progresses. Scrum projects can provide a high level of progress visibility by

* Placing a high value on open, honest communication among the scrum team, stakeholders, customers, and anyone else within an organization who wants to know about a project.
* Daily scrums that provide daily insight into the development team’s immediate progress and roadblocks.
* Daily scrums around task boards enable developers to self-organize and identify the highest-priority tasks for the day.
* Using the information from daily scrum meetings, sprint burn-down charts, and task boards allows the project team to track progress for individual sprints.
* Sprint retrospectives allow scrum team members to identify what’s working well and what’s not to make action plans for improvement.
* Demonstrating accomplishments in sprint reviews. Anyone within an organization may attend a sprint review, even members of other scrum teams.

**Increased project control**

Scrum teams have numerous opportunities to control project performance and make corrections as needed because of the following practices:

* Adjusting priorities throughout the project at each sprint interval rather than at major milestones allows the organization to have fixed-time and fixed-price projects while accommodating change.
* Embracing change allows the project team to react to outside factors like market demand.
* Daily scrum coordination allows the scrum team to quickly address issues as they arise, and swarm together to get requirements to done.
* Daily updates to sprint backlogs mean that sprint burn-down charts accurately reflect sprint progress, giving the scrum team the opportunity to make changes the moment it sees problems.
* Face-to-face conversations remove roadblocks to communication and issue resolution.
* Sprint reviews let project stakeholders see working products and provide product owners the feedback they need to ensure that the project stays on track.
* Sprint retrospectives enable the scrum team to make informed course adjustments at the end of every sprint to enhance product quality, increase development team performance, and refine project processes.

The many opportunities to inspect and adapt throughout scrum projects allow all members of the project team — the development team, product owner, scrum master, and stakeholders — to exercise control and ultimately create better products.

**Reduced risk**

Scrum helps mitigate the risk of absolute project failure — spending large amounts of time and money with no return on investment — by delivering tangible product early and forcing scrum teams to fail early if they’re going to fail at all through the following practices:

* Having the highest-risk items done first provides the longest runway to work through issues or fail early and inexpensively.
* Developing in sprints, ensuring a short time between initial project investment and either failing fast or validating that a product or an approach will work.
* Having a working product increment starting with the very first sprint, so that even if a project gets terminated, the highest-value and risk requirements have been developed and could be delivered to the customer if desired.
* Developing requirements to the definition of done in each sprint so that project sponsors have completed, usable features, regardless of what may happen with the project in the future.
* Providing constant feedback on products and processes.

2. Based on the Agile Manifesto

* 1. Individuals and interactions over processes and tools.
  2. Working software over comprehensive documentation.
  3. Customer collaboration over contract negotiation.
  4. Responding to change over following a plan.