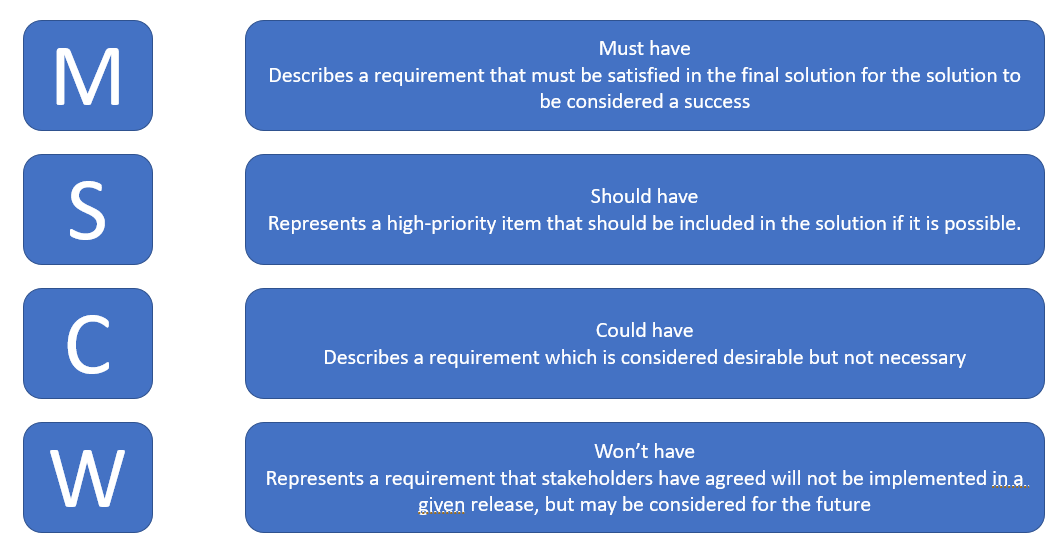
**What is MoSCow technique?**

The MoSCoW method is a prioritization technique used in management, business analysis, project management, and software development to reach a common understanding with stakeholders on the importance they place on the delivery of each requirement; it is also known as MoSCoW prioritization or MoSCoW analysis. MoSCow Prioritisation Technique plays a key role in Agile Project Management. In an Agile project it is vital to understand the importance of different things. This is because time is a fixed resource, so prioritisation is applied to requirements, tasks, products, user cases, etc. Be very careful, this is not Agile technique (include Scrum, Kanban, XP and so on) however this technique is widely used in Agile. There are some questions in almost all certifications regarding MoSCoW technique.

The term MoSCoW itself is an acronym derived from the first letter of each of four prioritization categories (Must have, Should have, Could have, and Won't have), with the interstitial Os added to make the word pronounceable. While the Os are usually in lower-case to indicate that they do not stand for anything, the all-capitals MOSCOW is also used.



**Prioritization of MoSCoW requirements**

All requirements are important, but they are prioritized to deliver the greatest and most immediate business benefits early. Developers will initially try to deliver all the Must have, Should have and Could have requirements but the Should and Could requirements will be the first to be removed if the delivery timescale looks threatened.

**Must Have**

These provide the Minimum Usable Subset (MUS) of requirements which the project guarantees to deliver. This may be defined using some of the following:

Cannot deliver on target date without this

No point in delivering on target date without this; if it were not delivered, there would be no point deploying the solution on the intended date

Not legal without it

Unsafe without it

Cannot deliver the Business Case without it

Ask the question, “what happens if this requirement is not met?” If the answer is “cancel the project – there is no point in implementing a solution that does not meet this requirement” then it is a Must Have requirement. If there is some way round it, even if it is a manual workaround, then it will be a Should Have or a Could Have requirement. Downgrading a requirement to a Should Have or Could Have does not mean it won’t be delivered, simply that delivery is not guaranteed.

**Should Have**

Important but not vital

May be painful to leave out, but the solution is still viable

May need some kind of workaround, e.g. management of expectations, some inefficiency, an existing solution, paperwork, etc.

A Should Have may be differentiated from a Could Have by reviewing the degree of pain caused by it not being met, in terms of business value or numbers of people affected.

**Could Have**

Wanted or desirable but less important

Less impact if left out (compared with a Should Have)

**Won’t Have this time**

One benefit of the MoSCoW method is that it places several initiatives in the “won't-not-have” category. This helps manage expectations about what will not be included in a specific release (or other time frame you’re prioritizing for).

Placing initiatives in the “won't-not-have” category is one way to help prevent scope creep. If initiatives are in this category, the team knows they are not to be a priority for this specific time frame. Some initiatives in the “won't -not-have” group will get prioritized in the future, while others are not likely to happen at all. Some teams decide to differentiate between those by creating a subcategory within this group.

**Summary**

MoSCoW (Must Have, Should Have, Could Have, Won’t Have this time) is primarily used to prioritise requirements, although the technique is also useful in many other areas. Atern recommends no more than 60% effort for Must Haves for a project, with 40% Shoulds and Coulds. Anything higher than 60% poses a risk to the success and predictability of the project, unless the environment is well understood, the team is established and the external risks are minimal.