

## MOSCOW TECHNIQUE - Includes Free Template

Requirements need to be prioritized because stakeholders can't always have everything they want, or should I say, because we can't always give them everything they want. This is not because we don't like their faces but because most projects are faced with a limited budget and time frame. As a BA, how do you ensure you focus on the most important requirements?

The MoSCoW technique is used by analysts and stakeholders for prioritizing requirements in a collaborative fashion.

Using a Human Resources System as an example, here's an explanation of the MoSCoW Technique:

### **MUST (M)**

Defines a requirement that has to be satisfied for the final solution to be acceptable e.g. The HR system "must" store employee leave history.

### **SHOULD (S)**

This is a high-priority requirement that should be included if possible, within the delivery time frame. Workarounds may be available for such requirements and they are not usually considered as time-critical or must-haves. e.g. The HR system "should" allow printing of leave letters.

### **COULD (C)**

This is a desirable or nice-to-have requirement (time and resources permitting) but the solution will still be accepted if the functionality is not included e.g. The HR system "could" send out notifications on pending leave dates.

### **WON'T or WOULD (W)**

This represents a requirement that stakeholders want to have, but have agreed will not be implemented in the current version of the system. That is, they have decided it will be postponed till the next round of developments e.g. The HR system "won't" support remote access but may do so in the next release.

You'll notice that the HR system features have been discussed in a decreasing order of priority - from what we must have, to what we should have, could have and won't have in that order.

### **Practical Application**

This technique is best used when the BA has gathered all existing solution requirements.

1. Assemble all stakeholders – Each stakeholder, with help from the BA, is responsible for assigning priorities to the requirements that fall under their purview
2. All Requirements may be listed on a flip chart and prioritised by assigning categories to each ([M, S, C or W](#)).
3. If there are multiple stakeholders with different opinions on what category to assign to a requirement, voting can be used to reach consensus.
4. Present categorized requirements in a readable format - [See template here](#)
5. The requirements should be reviewed throughout the project as stakeholder needs may evolve with time.

The BABOK Guide provides 8 criteria to be used for assigning priorities to requirements. They are:

- **Business Value:** Which requirement provides the most business value? The more business value a requirement will deliver, the greater the priority stakeholders may choose to assign to it.
- **Business or Technical Risk:** Some requirements pose a significant risk of project failure if not implemented successfully. The analyst may assign a high priority to this category of requirements so that if the project does fail, the least amount of effort would have been spent.
- **Implementation Difficulty:** Which requirements are the easiest to implement? Straightforward requirements may lead to quick-wins and provide an opportunity for the project team to familiarize themselves with other elements of the project before taking on more complex requirements for implementation.
- **Likelihood of Success:** Which requirements can provide quick-wins and increase the probability of project acceptance? If the objective of the project is to demonstrate value as quickly as possible and quell negative chatter, requirements that have a higher probability of success would be given high priority.
- **Regulatory Compliance:** Which requirements are necessary for policy and regulation adherence? These requirements are non-negotiable and usually have to be implemented within a set period of time, causing them to take precedence over stakeholder requirements in some cases.
- **Relationship to other requirements:** Which requirements support other high-value business requirements? Such requirements may be assigned a high priority because of their link to important requirements.
- **Urgency:** Which requirements have a high degree of urgency? Most stakeholders tend to place a high priority on requirements needed like yesterday
- **Stakeholder Agreement:** Requirements on which stakeholders disagree should be postponed or assigned a low priority until consensus can be reached on their usefulness.

The above criteria can be used for prioritizing requirements by assigning weights to each requirement using a decision table. Requirements with the highest scores receive greater priority than those with lower scores. Other useful techniques for requirements prioritization are *time-boxing* and *voting*.

MoSCoW technique was introduced by Dai Clegg of Oracle UK in 1994.

# MoSCoW Prioritisation

## 1 Introduction

In an Atern project where time has been fixed, understanding the relative importance of things is vital to making progress and keeping to deadlines. Prioritisation can be applied to requirements, tasks, products, use cases, user stories, acceptance criteria and tests. MoSCoW is a technique for helping to understand priorities. The letters stand for:

- **Must Have**
- **Should Have**
- **Could Have**
- **Won't Have this time**

The reason we use MoSCoW in Atern is that the problem with simply saying that requirements are of High, Medium or Low importance is that the definitions of these priorities are missing. Using MoSCoW means that priorities are specific. The specific use of Must, Should, Could or Won't Have implies the result of failing to deliver that requirement.

## 2 The MoSCoW Rules

These are some possible definitions of what the different priorities mean. It is important to agree the definitions with the users. Preferably this is agreed before the requirements are captured – i.e. before it becomes emotive.

### 2.1 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.

## **2.2 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.

## **2.3 Could Have**

- Wanted or desirable but less important
- Less impact if left out (compared with a Should Have)

## **2.4 Won't Have this time**

These are requirements which the project team has agreed it will not deliver. They are recorded in the Prioritised Requirements List where they help clarify the scope of the project and to avoid being reintroduced 'via the back door' at a later date. This helps to manage expectations that some requirements will simply not make it into the delivered solution, at least not this time around.

# **3 Ensuring effective prioritisation**

## **3.1 Agreeing how priorities will work**

Prior to requirements capture, the definitions of Must Have, Should Have, Could Have and Won't Have need to be agreed with the business. Some examples are described above. However, the Must Have definition is not negotiable. Any requirement defined as a Must Have will have a critical impact on the success of the project. The Project Manager or Business Analyst should challenge requirements if they are not obvious Must Haves; it is up to the Business Visionary or their empowered Business Ambassador to prove a requirement is a Must Have. If he/she cannot, it is a Should Have at best.

Agree escalation or decision-making processes, e.g. Business Ambassador to Business Visionary to Business Sponsor, and agree the level of empowerment around decision-making at each level.

\At the end of an increment, all unsatisfied requirements are reprioritised in the light of the needs of the next increment. This means that, for instance, a Could Have that is unsatisfied in an increment may be demoted subsequently to a Won't Have, because it does not contribute enough towards the business needs to be addressed next.

### **3.2 The Business Sponsor's perspective**

The MoSCoW rules have been cast in a way that allows the delivery of the Minimum Usable Subset of requirements to be guaranteed. Both the team and those they are delivering to can share a confidence in this because of the high degree of contingency allowed in the delivery of the Must Haves. A rule of thumb often used is that Must Have requirements do not exceed 60% of the effort. If this rule is followed, then that ensures contingency represents at least 40% of the total effort.

So is this all that the Business Sponsor can expect to be delivered? The answer is an emphatic "No". Whilst understanding that there is a real difference between a guarantee and an expectation, the Business Sponsor can reasonably expect more than this to be delivered except under the most challenging of circumstances. This is where the split between Should Haves and Could Haves comes into play.

If the Should Haves and Could Haves are split evenly with 20% of the total effort associated with each then the Musts and Shoulds, in combination, will represent no more than 80% of the total effort. The remaining 20% of effort associated with the Could Haves is now the contingency available to protect the more important requirements. By most standards this is still a very reasonable level of contingency and rightly implies that the Business Sponsor can reasonably expect the Should Have requirements to be met. It is just that, quite understandably, the team does not have the confidence to make this a guarantee. So, sensible prioritisation, combined with timeboxing leads to predictability of delivery and therefore greater confidence. Keeping project metrics to show the percentage of Should Haves and Could Haves delivered on each increment or timebox will either re-enforce this confidence, if things are going well, or provide an early warning that some important (but not critical) requirements may not be met if problems arise.

### **3.3 MoSCoW and the Business Case**

The best way to address prioritisation initially is with a quantified Business Case. This should support Feasibility and be revisited during Foundations. If a Business Case does not exist, the Business Sponsor and Business Visionary need to articulate the business drivers, preferably in a quantified form. Some practitioners believe that any requirement contributing to the Business Case should be defined as Must Have, others accept that a small reduction in benefit is unlikely to make a project completely unviable and desire a more pragmatic solution. These practitioners believe that it is sensible to allow the requirements contributing to the Business Case to span Must Have and Should Have requirements.

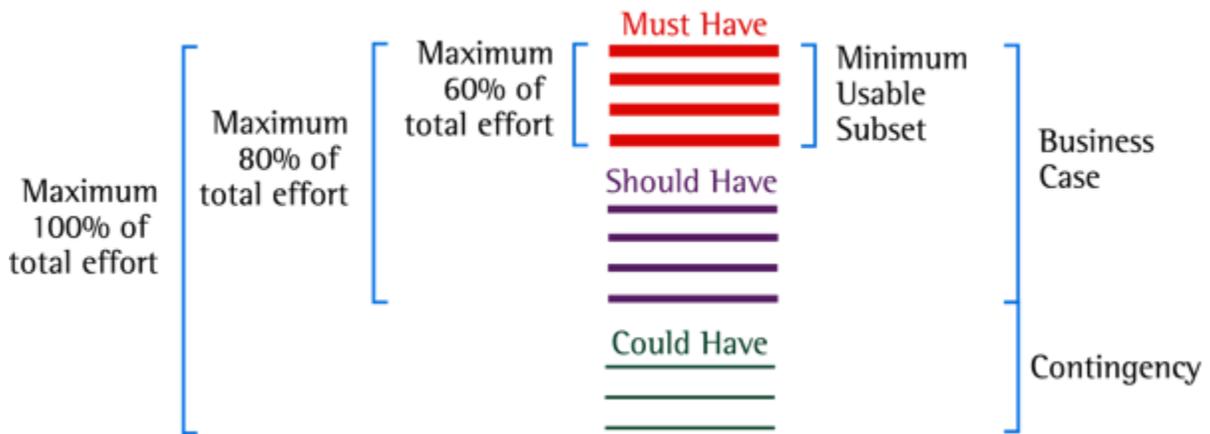


Figure 10a: MoSCoW and the Business Case It is likely that contractual relationships (whether formally between organisations or informally within an organisation) will influence the decision on this issue one way or the other.

#### 4 Levels of priority

MoSCoW prioritisation is really only meaningful in a specified timeframe and the same requirement may have a different priority in that context. A Must Have requirement for the project as a whole may not be a Must Have for the first increment. For example, even if a Must Have requirement for a computer system is the facility to archive old data, it is very likely that the solution could be used effectively for a few months without this facility being in place. In this case, it is sensible to make the archive facility a Should or a Could Have for the first increment even though delivery of this facility is a Must Have before the end of the project. Similarly, a Must Have requirement for an increment may be included as a Should or a Could Have for an early Development Timebox. Many consider this approach to be sensible as it allows the more important requirements to be addressed earlier rather than later but, if taking this approach, beware the risk of confusion. Each deliverable effectively has two or even three priorities in different timeframes and the Project Manager needs to ensure that the team do not lose sight of the real business priorities. The best way to deal with this is

to create a Timebox PRL, a subset of the Project PRL that is specifically associated with a timebox and leave the priorities unchanged on the main PRL for the project.

## **5 What to prioritise**

Every item of work has a priority. Priorities are set before work commences and kept under continual review as the work is done. As new work arises either through introduction of a new requirement or through the exposure of unexpected work associated with existing requirements, the decision must be made as to how critical they are to the success of the current work using the MoSCoW rules. All priorities should be reviewed throughout the project to ensure that they are still valid.

## **6 How many of each priority?**

When deciding how much effort should be Must Have requirements, bear in mind that anything other than a Must is, to some degree, contingency. The aim is to get the percentage effort for Must Haves (in terms of effort to deliver) as low as possible and to be wary of anything above 60%, i.e. 60% Musts Haves, 40% Should Haves and Could Haves. Won't Haves are excluded from the calculation, as they won't be part of this project/increment/timebox. Levels of effort above 60% for Must Haves introduce a risk of failure, unless the team are working in a project where estimates are known to be accurate, the approach is very well understood and the environment is understood and risk-free in terms of the potential for external factors to introduce delays.

## **7 Hierarchies of priorities**

Requirements are identified at various levels of detail, from a high-level strategic viewpoint (typically at Feasibility) through to a more detailed, implementable level (typically during Exploration and Engineering). High-level requirements can usually be decomposed and it is this decomposition that can help resolve one of the problems that confront teams: all requirements appear to be Must Haves. If all requirements really were Must Haves, the flexibility derived from the MoSCoW prioritisation would no longer work. There would be no lower priority requirements to be dropped from the deliverables to get the project back on time and budget. In fact, this goes against the whole Atern ethos of fixing Time and Resources and flexing Features (the triangles diagram).

Believing everything is a Must Have is often symptomatic of insufficient decomposition of requirements. A high-level Must Have requirement frequently yields a mix of sub-

requirements, each with a different priority. Flexibility is once more restored and some of the detailed functionality can be dropped from the delivered solution so that the project deadline can be met. Where a requirement has a Must Have below a Should Have for example, this would signify that if this requirement were to be delivered, it must have the lower level requirement to be acceptable.

## **8 Tips for assigning priorities**

1. Work closely with the Business Visionary to ensure they are fully up to speed as to why and how Atern prioritises requirements.
2. Start all requirements as Won't Haves and then justify why they need to be given a higher priority.
3. For each requirement that is proposed as a Must Have, ask: "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.
4. Ask: "I come to you the night before deployment and tell you there is a problem with a Must Have requirement and that we can't deploy it – will you stop the deployment?" If the answer is "yes" then this is a Must Have requirement.
5. Is there a workaround, even if it is manual? If there is, then it is not a Must Have requirement. Compare the cost of the workaround with the cost of delivering it, including the cost of any associated delays.
6. Ask why is the requirement needed – for this project and this increment.
7. If there is a Business Case in sufficient detail, can it be used to justify the intended priority? If not, create one.
8. Is there more than one requirement implied in a single statement? Are they of the same priority? Decompose the requirement!
9. Is this requirement dependent on any others being fulfilled? A Must Have cannot depend on the delivery of anything other than a Must Have because of the risk of it not being there.
10. Allow different priorities for levels of acceptability of a requirement. For example. "The current back-up procedures will be followed to ensure that the service can be restored as quickly as possible." How quick is that? Given enough time and money, that could be within seconds. They may say that it Should happen within four hours, but it Must happen within 24 hours, for example.
11. Can this requirement be decomposed? Is it necessary to deliver each of those components to fulfil the requirement? Are the decomposed elements of the same priority as each other?

12. Tie the requirement to a project objective. If the objective is not a Must Have, then probably neither is the requirement relating to it.
13. Remember that team members may cause scope creep by working on the fun things rather than the important things. MoSCoW can help avoid this.
14. Does the priority change with time? For example, for an initial phase, it is a Should Have but it will be a Must Have for the second increment.
15. Prioritise defects/bugs, using MoSCoW.
16. Prioritise testing, using MoSCoW.
17. Use MoSCoW to prioritise your To Do list. It can be used for activities as well as requirements.

## 9 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.**

Top Prioritisation Tips:

- Agree what the priorities mean early in the project
- Use all the priorities
- Challenge Must Haves
- Control the number of Must Haves
- Prioritise everything – it helps the concept become deeply ingrained in the team's approach.