# 9 principles for successful software implementation

SaaS Adoption • Change Management

Whether it's collaborating remotely, keeping up with their compliance workload, or making sense of complex, unstructured data, ever more businesses are turning to tech to solve their problems. Gartner reckons global tech spending will hit $3.9 trillion (approximately £3.2 trillion) in 2020. And most of it will be spent on software.

But while the right software can make your staff's lives easier, give you a competitive edge, and save you money in the long run, integrating it into your organisation isn't always smooth going. It can take time to start seeing its value, and this is often down to implementation issues.

So how can you overcome these challenges and reap the rewards of your investment as quickly as possible?

## The 3 stages of software implementation

Successful software implementation projects broadly follow three stages:

**Discovery**

This is where you dive deep into your problem and think up ways you could solve it

**Configuration**

Here, you set up your chosen software so it addresses your needs

**Adoption**

This is where end-users get access to the software and start using it

Most projects run into issues during adoption.That said, software implementation has many moving parts, and all the pieces matter. So, the root cause of adoption issues can often be traced back much earlier — to configuration stage, or even discovery stage.

With this in mind, we've developed 9 principles we follow to help set up every software implementation project for success.

Here's an in-depth look at them.

## 1. Start with your goals

There are enough goal-setting techniques out there to spell the whole alphabet several times over: SMART, ABC, WOOP... the list goes on.

It doesn't matter which goal-setting method you pick as long as you're comfortable wielding it to your advantage. But what's important is that your goals:

- Address your business objectives
- Address your end-users' objectives
- Are measurable

Let's say you want to reduce non-compliance incidents across your business.

For the leadership team, that's the end goal. But to hit that goal, the compliance team will need the software to do certain things.. For example, they may want it to be able to slash their admin, so they can spend more time on staff training.

This means your goals shouldn't just reflect your aspirations as a business. They must also reflect the practical realities of those who will ultimately do the work that helps you get there. Otherwise, they'll be much harder to achieve.

More importantly, your goals should help you clearly define what success would look like. Which brings us to the next point.

## 2. Set effective metrics

If hitting your goals proves your project was successful, metrics show you how close you are to getting there.

Effective metrics should help you:

- Track your progress
- Evaluate whether your chosen platform is getting you where you want to be

It goes without saying, but the clearer and more measurable your goals are, the easier this step will be.

Let's say you want to cut admin by 30% within six months. The obvious metric here would be time spent on specific tasks. So you could measure how long the tasks take to complete before you implement the software, and measure again at regular intervals once you've deployed it.

If your goals are harder to measure, it may be helpful to think of indirect indicators. So if you want to increase your staff's understanding of complex policies, you could measure how often they consult your policies today, and check whether they do so less after you deploy the software.

## 3. Get end-users on board early on

GRC expert Michael Rasmussen notes that there's often a disconnect between how business leaders think things are going and what is happening on the ground. While leaders might believe the new software is the best thing since sliced bread, end-users may be scrambling to find workarounds so they can get things done.

More often than not, this is because end-users were involved too late in the process.

Nobody understands your day-to-day realities better than those who contend with them every day. So it's important to get their perspectives early on.

At KV Labs, we typically hold one-on-one interviews where we break down:

- Each individual's role and job
- The tasks they have to perform every day
- The problems they encounter while performing their tasks
- Why those problems happen

Of course, not every interview will be insightful. But listening, staying curious, and asking why will show staff you intend to make their life easier and, hopefully, get them invested in the project's success.

## 4. Pilot first, scale later

As TSB and others know all too well, deploying too fast too soon can spell disaster. The better, safer way is to start small and build on your successes. This will allow you to keep things manageable as you iron out teething problems.

That said, using a pilot purely to find out whether software can do what you were told it could do is limiting. It's worth treating it as an opportunity to experiment. This not only helps you build your business case, but may also uncover use cases you might not have considered.

You should also make your pilot as representative as possible so it mimics real-life scenarios. So while internal audit should definitely be involved in trialing an audit tool, don't forget risk, compliance, and other stakeholders.

Make a few trial runs and ask everyone to give feedback. What do they like about the software? And what problems are they experiencing?

These insights should be the backbone of your configuration.

## 5. Figure out your priorities

Because most software programmes are highly customisable, there will inevitably be competing priorities at configuration stage. These can quickly make a project go off the rails, so it's important to have a framework that helps you keep sight of your goals.

The key is to evaluate objectively, so people won't feel personally attacked if their ideas are rejected. Questions worth asking yourself include:

- Is this feature necessary for our goals?
- Will it open up more functionality?
- Will it increase sales?
- Will it increase adoption across the organisation?

Once you've gone through this checklist, it's also worth asking yourself how much effort is involved. You can then create a spreadsheet or use a Kanban-style app like Trello to keep track of each idea's priority level and the effort required.

## 6. Make sure you foster the right culture

Getting people to collaborate is probably the hardest part of any project. Ultimately, your people have to put plans in action. And if your company is siloed or office politics get in the way, that's going to be an uphill struggle.

Involving people early on, listening to their concerns, and getting them invested in your project is half the battle.

That said, it's also important to have a clearly defined structure. Everyone who's part of the project should know exactly what their role and responsibilities are. And someone must take ownership.

More importantly, you should take steps to foster a positive culture. Politics in particular can become toxic and hamper not just your current project, but also future attempts at innovation. Ben Horowitz's book The Hard Thing About Hard Things is chock-full of practical advice on creating a supportive work environment.

## 7. Get your messaging right

Once you've configured the platform, it's time for adoption. And for this to go smoothly, you need a strong, consistent message.

Your end-users should:

- Be aware they can now use the platform
- Get trained
- Know who to talk to if they need help or support

Your training should be tailored to your end-users. So while a general workshop might be enough for a relatively straightforward platform like Microsoft Teams, it won't work for a product that has a complex workflow and multiple touchpoints.

It also helps to make training as interactive and engaging as possible — think workshops, quizzes, and other activity-based learning. This will encourage people to get involved and help the information stick.

## 8. Collect feedback from power users

In an ideal world, 20% to 30% of your end-users will pick things up quickly, 60% will need some training, while 10% to 20% may need ongoing support.

It's a good idea to involve those 20% to 30% — or power users — in the decision-making process. They'll provide you with valuable feedback and help adoption along by supporting those who are having a harder time getting to grips with the platform.

## 9. Monitor your progress

This is where you should put the metrics in step 2 to work. Are people using the platform? Is it going the way you envisioned at discovery stage? And how close are you to achieving your goals.

It's worth spending time evaluating what's really going on, because progress may be slow for a variety of reasons. This will allow you to address issues effectively.

Let's say end-users are resistant to change. You could run additional training workshops to get people more comfortable with the platform.

By contrast, if the issue is a software limitation, further configuration may be able to solve it.

## Preparation goes a long way

Choosing, configuring, and integrating new software into your organisation can be surprisingly challenging, especially if it'll change the way people work. But if you carefully define what success should look like, keep both ears to the ground, and give your end-users the support they need, you'll be well on your way to making a success of it.

*Need help identifying the software that's best-suited to your needs, configuring it the right way, and deploying it within your organisation successfully?*

*Let's chat about how KV Labs can help.*