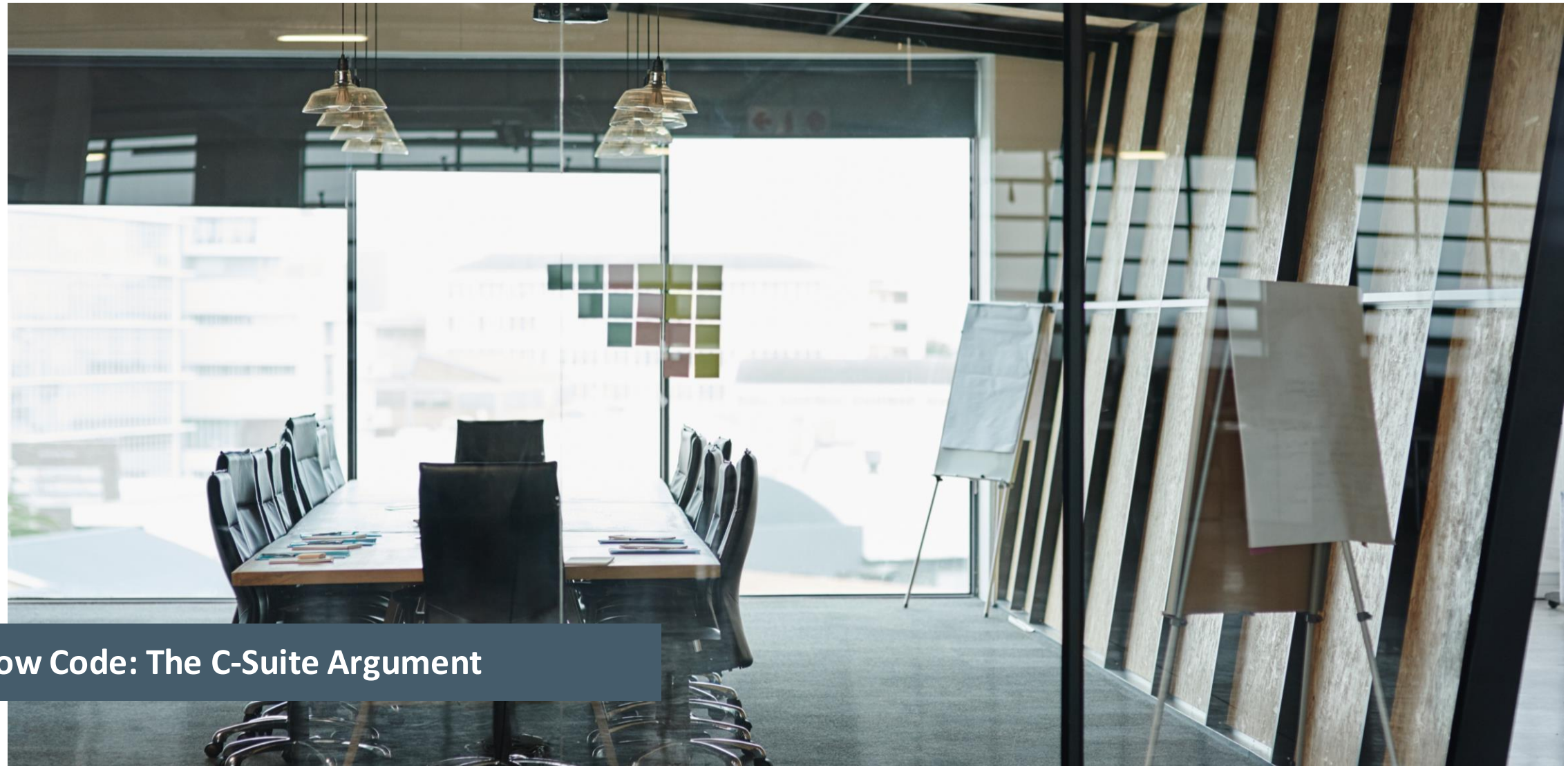


Low Code: The C-Suite Argument

The C-Suite Argument For Low-Code

Legacy approaches to systems hindering future progress

Historically organisations have had one of two options when building the foundational systems to drive their organisations :-

- **Buy a package "off the shelf"**

Most organisations do not have the investment capability nor the expertise to effectively build their own solution. As a consequence off-the-shelf packages have been the only option which bring their own set of compromises :-

Base configuration
How out of the box are the systems really, configuration can be as complex as building from scratch.

Deliver market differentiation
How do you stand out in the crowd if everyone is using the same underlying product? Features end up being offline or expensive spend with the vendor to bespoke.

Bespoke away from the main product
Tailoring the product to work within the organisation can lead to being locked to a current version and not able to take regular updates or require an expensive upgrade regression programme of its own.

- **Build from scratch**

Where organisations have a historic systems build capability the tendency is to build everything internally with the mistaken believe that avoiding annual licencing is the most cost effective way forwards. Where these systems are core to the business offering then this can make a lot of sense but we regularly see the same approach applied across all of the systems estate with the following consequences :-

Poor User Experience	Focus is on the functionality and not usability
Extended Support Costs	Requirement to schedule in emergency development for cyber events
End Of Life	The need to invest development time due to various parts of the development stack going out of support
Endless Prioritisation	Development time is expensive so the nice to haves never get dealt with

Low-code offers a middle ground which addresses the weaknesses of the above with little compromise.

The C-Suite Argument For Low-Code

Where should investment be directed?

What is the DNA of your IT development team – is it to build software or deliver business outcomes?
One of the regular conversations we have with the C-suite is that the IT function was there to enable the business, building software was just one way of enabling this and not the core *raison-d'etre*.

One such client got this and would re-use / off-the-shelf anything and everything to get cost effective (not cheap & nasty) solutions to enable the business. Software development was a relatively small part of the output, configuring and integrating was the primary role from the trade floor to operations. The strategic hierarchy was re-use, if you can't do that then buy, and if that isn't an option, only then build.

Is the IT spend mapped appropriately to the business priorities?
We ask our clients what their USP (Unique Service Proposition) is for their business and their business strategy…. Investment of $'s in bespoke software & intellectual property should be mapped to the outcomes that are truly distinguishing for the business and enable the business to deliver on the strategic goals. This is why Google never built a General Ledger, they were technically perfectly capable but it didn't add to their core business so was viewed as a distraction and diluting to their talent pool.

Business Process mapping and discipline?
Are the organisations' processes mapped from the top down into a process hierarchy / nomenclature / ownership / strategic alignment / customer experience / cost / time / quality – not down to every step of the process but so the playing field is understood and spend appropriately focused. Are the processes overlapping or conflicting when looked at sufficiently end to end, is there stupid stuff going on in there that should just be stopped? Who makes the decision whether an exception should be granted to an underwriting flow so it follows a bespoke path for a particular business?

Keep business logic in one place
This isn't really down to which workflow tool but in the business process discipline to enforce the business logic is only built once and not replicated or federated. This doesn't mean only one system, just that is it only built once where-ever it is built. This reduces the change friction for future build outs and simplifies tracking down issues.

The C-Suite Argument For Low-Code

How do I maximise the value delivery of my systems spend?

Can you do it better than a software house?
Would you run a data centre better than Amazon? Same principles apply for business process management – you are never going to build a better outcome that is more flexible and cost effective than the people that do this and spread the investment cost over 1,000's of customers. A tailored user interface is pure vanity when 90% of the required functionality can be delivered at 10% of the cost. Any monthly licencing costs will be dwarfed by the run-the-bank costs of engineers for bespoke built software.

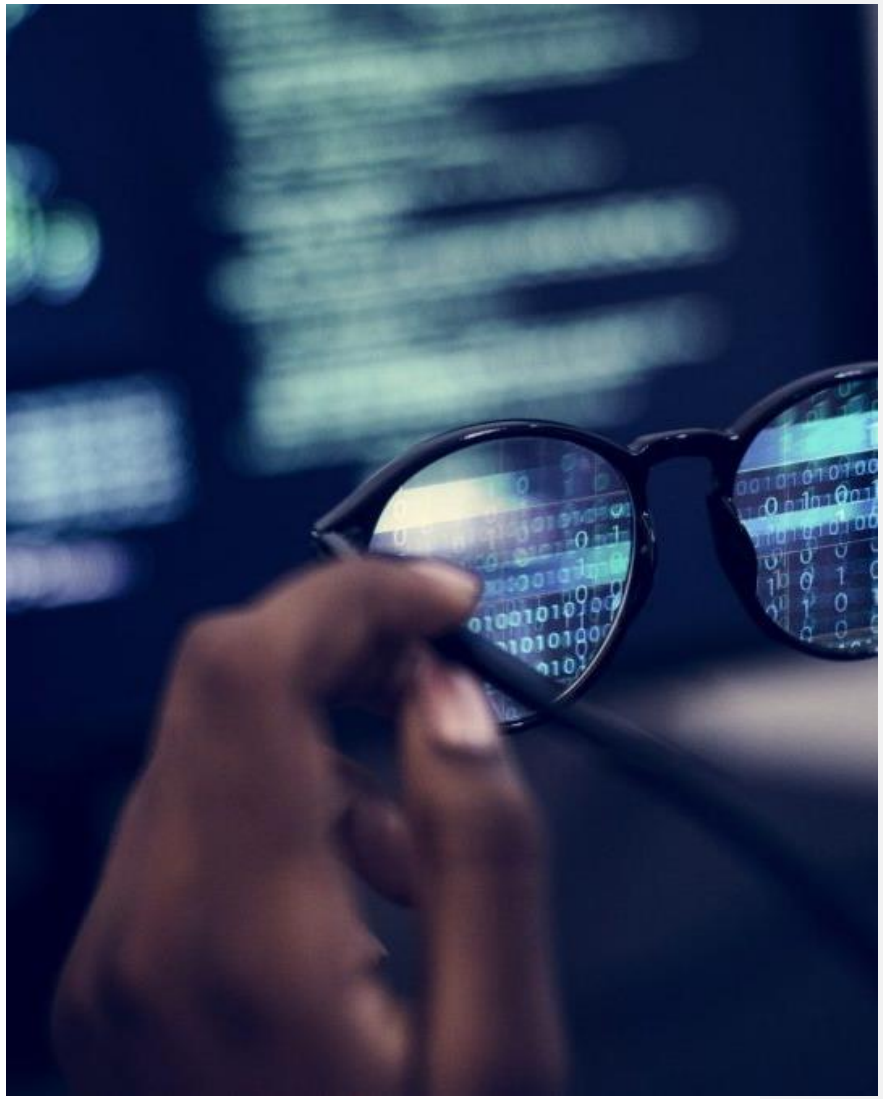
Move at pace?
Developing excellent software takes time and requires skilled engineers to get right. This can be judged to be fine for the core business value add tasks but inappropriate to laden the cost, time and inflexibility across all processes in the organisation. Sacrificing some ultimate flexibility will allow pods to burn through automations and digitisations moving the whole organisation onto a digital footing.

Focus on an area and build out internal confidence
Taken from the previous experience playbook, start with a friendly and enthusiastic area that can be used to both prove the benefits and flush out the issues of integrating with the organisation – and show how compelling it is. It's a slower but stickier approach and will get to the end game quicker with a better outcome.

Prioritisation so nothing other than key strategic gets done?
The end of this movie has been seen before many times – build out a strategic centralised tool (or tool set) and stop all other competing work to focus on it. The trouble then comes that nothing else other than core strategic gets done, all the other secondary and supporting processes continue as tactical because, by definition, they are not a core priority and repeatedly get trumped in the prioritisation debates.

No matter how many people you put in the centralised team, and at a point that will become counter-productive, sorting out the annoying processes that cost 1hr a month will never get done so you end up with permanent organisational and change friction at the cost of the employee experience.

The C-Suite Argument For Low-Code



At NextWave we offer a full solution set to help with the transition to low code and have help clients with the following strategic engagements :-

- Building business cases to support the strategic investment in Business Process Management and low-code
- Broad stake-holder engagement to build the business support
- Identifying the most appropriate candidates and structuring books of work
- Reviewing and challenging strategic approaches and proposing challenger options
- Integrating strategies across process, data and people

At NextWave we also have a team of dedicated engineers who can turn these strategies into reality whilst enabling organisations :-

- Architectural and strategic foundational requirements for low-code
- Assisting the with onboarding and enabling
- Delivering business outcomes whilst mentoring internal teams
- Training and education for internal upskill
- On retainer support and assistance