Patrick Gormley is Head of Capita Consulting with experience including management level roles at IBM, Capco and Accenture.

Oli Freestone heads up our Institute, which is part of Capita's digital and consulting capability and is responsible for digital thought leadership, research and insights. He has worked across multiple industries as a management consultant, and has expertise in strategy, technology and innovation, and is a regular contributor to leading publications on these topics.

Marc Hollyoak is an Associate Partner and CTO at Capita Consulting with years of experience in exploring customer experience, value proposition, and overarching business models. Using design thinking and lean start-up approach, collaborating and leveraging emerging technology, to deliver a significant increase in customer value and business growth.

Tom Heywood is part of Capita's digital and consulting capability, helping clients in both the public and private sector. Tom comes from a background in engineering, consultancy and solution design, and helps create better outcomes by solving problems - whether through redesigning services, developing new operating models or creating new propositions.

Coding: Less is More

When it comes to code, less is more… much more!

If you've just signed up to a multi-year deal to build a large-scale digital factory you might want to stop reading… the days of throwing bodies at software development are numbered.

However, you could almost be forgiven as you were probably just following the herd… after all 'Software Engineer' is the most in demand profession in the world[1].

Tech companies and delivery partners parade the number of engineers around like a badge of honour – a thousand strong here, two thousand heads there. The idea of throwing bodies at software projects has become a bit like the outsourcing models of yester year – think about cost and use labour arbitrage to drive out short term savings.

In fact, India, once the darling of the BPO (Business Process Outsourcing) market, now has the fastest growing software development base in the world, estimated to be c. five million by the end of this year[2].

In the US, employment of software developers is projected to grow 24% from 2016 to 2026, much faster than the average for all occupations[3]. This had led to the median salary for a software engineer in the US to grow to over $100,000[3], whilst in Silicon Valley a lead engineer at a top tech firm can expect to make twice this figure.

But this "bigger is better / more is faster" approach applies lazy thinking to a problem that needs radical rethinking. Instead of being Gillette and adding another blade to the tool, the whole tool needs to be stripped down and rethought.

Think smarter

The smarter thinkers are doing just that, and reimagining development using a low-code / Rapid Application Development (RAD) approach that asks how can we get to an outcome faster by re-using components and building blocks which have already been built? What if instead of adding more lines of code we re-purposed blocks we'd already built, or better still used someone else's?

Capita

The leaders in this field are, unsurprisingly, the newer kids on the block. The Ubers of this world have gone as far as to engineer out proprietary code from large sections of their own applications, instead choosing to use microservices architectures and APIs (Application Programming Interface) to hook into other services, both internally and externally (think Google for maps, Twilio for messaging and Braintree for payments[4]).

There are even examples of 'no-code' platforms emerging, where companies have white-labelled the entire stack and poured their efforts into brand and marketing.

So how does RAD impact outcomes?

Well, remember all those people who told you that you couldn't break the 'iron' triangle of project management – that you couldn't have high quality, at speed, and cheap? They didn't work in RAD.

Incidentally the iron triangle was conceived in the 1950s shortly after IBM's then leader, Thomas J. Watson Jr, predicted that the world market for computers would be five units, so is due an upgrade!

Here's how it improves outcomes for end users:

Better

RAD allows the end user to steer the solution design more closely to meet their outcomes, as the solution evolves fast (we're talking in front of their eyes), the end user can try it out in practice and give real time feedback. Not what you expected? Change it. Need to beef up a feature? Improve it. Want to connect to another service? No problem.

For the non-techy business user who glazes over when faced with screens full of Python or Ruby, RAD and low-code approaches also improve the experience hugely – connecting microservices using APIs is intuitive an can be done via WYSIWYG interfaces that make sense to the non-technical audiences. Think connecting to Google Maps plot search results as markers on a map, or accepting payments from customers via Stripe.

In addition, the approach helps to engineer out bugs quicker because there's minimal bespoke, un-tested code to deal with, and the bugs that do occur can be addressed faster.

Faster

Digital natives implicitly expect hundreds of micro-releases a day – digital services now 'morph' like water in response to the thousands of automated experiments running across the platforms 24/7/365. Amazon famously deploys new software every 11 seconds and has been clocked at over 1000 deployments an hour[5].

By treating services as componentised blocks, they're able to spin up services really fast (we're talking days and weeks rather than months). And with that comes big cost savings (some estimates suggest up to 90% - 95% saving on a typical large-scale bespoke app build[4]).

Cheaper

The solution is very lean and therefore by design more cost effective. This means less friction and less feature bloat because the end user is involved in the complete process and can literally see it building before their eyes – no long weeks of design workshops, no black box development, no big reveal – they pay less to get a solution that precisely tracks and meets their (changing) needs.

And for digital service providers?

Better

As a result of more abstraction, providers can now move from computer-centric to human-centric development, allowing them to focus on real value generating features rather than getting sucked into deep technical debates.

With a low-code platform, all the security, cross-platform support and data integration capabilities have already been built and can be customized easily. Developers can focus on solving business problems rather than mundane, error-prone technical requirements. The risk of catastrophic failure drops significantly.

And because low-code platforms build and update apps so quickly, developers can also now share fully functional features with stakeholders in days or even hours, resulting in much faster feedback cycles and designing out those frustrating 'in-between' periods whilst you cross your fingers hoping your provider has understood your ask. Maintaining a short, dynamic iteration cycle fosters more active engagement. Also, there is a clear, ongoing sense of progress and responsiveness.

Plus, you can really deliver on innovation and impact (and not have customers wince as if you were selling them the latest brand of snake oil). Hand coding an app is time-consuming and labour-intensive work. With low-code platforms, developers can create innovative new functionality or customize features without the mundane process of coding dragging them down. In addition to its direct benefits, this shift tends to keep top performers more engaged and enthusiastic, which can help you maintain a more satisfied, stable team.

Finally, complex governance can be a thing of the past. Historically when IT backlogs grew, business leaders would outsource work to external vendors. This could address short-term needs, but it often created longer-term challenges such as the need to tightly manage a multi-vendor landscape. Multiple IT teams and projects would become increasingly difficult to manage. By shortening lead times and using common infrastructure, low-code platforms can enhance transparency, security and compliance and take the pressure away from application governance.

Faster

RAD and low-code brings Dev and Ops together to enable a change to be published live many times a day. Whilst not many shops will deliver Amazon-like performance from day one, the discipline of pushing to live regularly forces developers to take ownership of their work and not just chuck it over the fence. This ‘full lifecycle’ approach to app development also means that it doesn’t end after a successful launch. After all, there are updates, feature improvements, and bug fixes to consider.

With a low-code platform, you can often make complex updates and deliver new features (such as integrating data from a third-party SaaS app) in minutes. And, it's done with minimal impact on other projects in the development queue.

Cheaper

As a result of more bang for the same buck (estimates at 2-5x productivity gain) and the ability to get continual user feedback, wasted effort is massively reduced. The solution is focussed on outcomes rather than process and is therefore leaner.

You also don’t need to pay through the nose for specialist development resource. The intuitive, visual editors that are part and parcel of low-code platforms do a lot to "level the playing field." Knowledge of specific languages isn't required, nor do they have to have years of experience to use them. Therefore, more developers can contribute to a project. In some cases, even non-technical stakeholders can learn to build their own prototypes. Waiting for IT support becomes a thing of the past.

In effect, you are going in with a ready-sharpened saw rather than spending hours preparing your tools. This keeps the solution leaner and lowers risk, which shortens the time to value and improves ROI.

The best low-code platforms are game-changers for those accustomed to traditional methods. Why? Because you can build apps for multiple platforms simultaneously. Not only that, but it can be done in a fraction of the time needed to hand-code an app for just one.

Software development, having been the engine room of disruption for years, now finds itself being disrupted – and the legacy incumbents that thought they had climbed out of trouble will find themselves hamstrung by the very structures and capabilities they’ve built to get them back in the game. Ironic.

RAD and low-code development is faster, better and cheaper – don’t follow the herd!

References

1. https://www.michaelpage.co.uk/news-and-research-centre/media-releases/software-engineer-is-the-worlds-most-in-demand-profession-new-analysis-reveals
2. https://www.theweek.in/news/biz-tech/2018/04/18/india-fastest-growing-software-developer-base-globally.html
3. https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm
4. https://www.cio.com/article/3250490/why-low-code-platforms-are-the-future-of-app-development.html
5. https://www.helpsystems.com/blog/6-companies-are-doing-devops-well

Capita