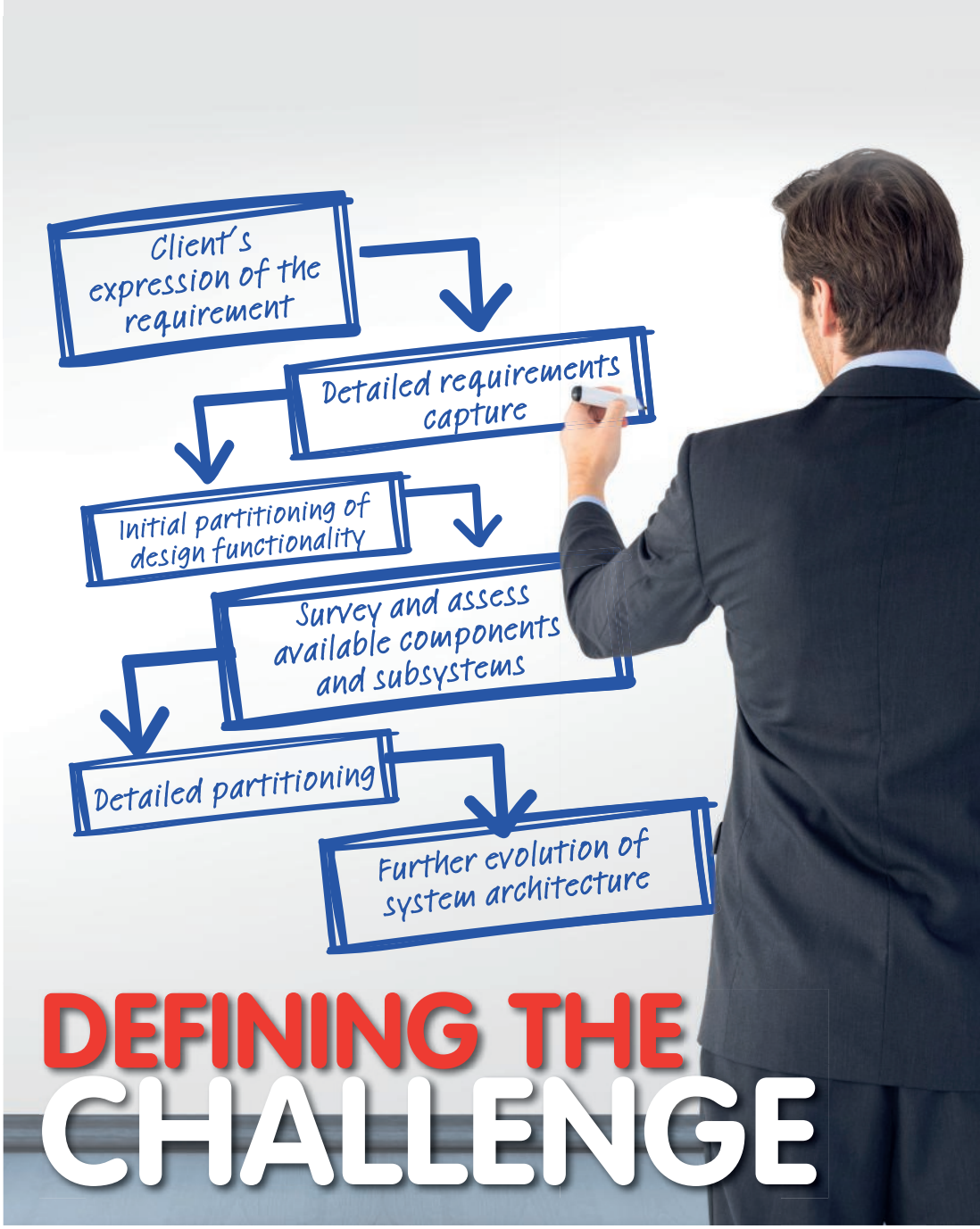
DEFINING THE CHALLENGE



When you are given an idea on the 'back of an envelope', what's the best way to work out a system architecture? By **Graham Pitcher.**

The emergence of the Internet of Things has also seen the emergence of a string of start ups which have, in their opinions, good ideas for IoT products, but who lack the technical expertise that would allow them to get those ideas to market.

One way for those companies to move forward is to work with a design consultancy; an organisation which, if necessary, can take that idea from the quintessential 'back of the envelope' all the way to manufacturing. But what does the consultant do when confronted with the germ of an idea?

Steve Greendale, principal consultant with Plextek, said what's typical in such situations is that a client has a problem which needs a solution. "Often, it's deriving an architecture for an end system, but sometimes it's just for part of a system. Tackling the problem is all to do with partitioning; deciding how best to split the solution into hardware and software elements."

Greendale noted that it would be helpful if the client had an idea of a potential solution, adding 'some kind of diagram is always useful'. "But this phase can be quite complicated when there's little in the way of external constraints."

Capturing requirements

One of the early steps in the process is to capture the requirements. While this step is probably no surprise, it helps to define the challenge. "What are the things that are 'must haves'?," Greendale asked. "What are the things that might be 'nice to have'? How many will be made? What is target manufacturing cost and what are the regulatory issues? All of these are important issues and, sometimes, the client may not have thought of all of these questions."

In general, Greendale observed, most systems will have areas where the desired behaviour is easy to map out. "Other areas may prove more difficult. These tend to be things that are best done in software. We would want to handle any behaviour that's decision drive in the software domain, simply to speed product development."

Greendale said that, once it has been decided to use a processor, it's important to make it 'pay for itself'. "We do this by mapping as much functionality as possible on to the processor."

He added that a further benefit of a processor based design is that developers can concentrate initally on developing code, leaving more peripheral areas until later in the process.

"Nevertheless," he stressed, "it's important to have placeholders. But having the design in the software domain allows you to concentrate on that 'must have' functionality." Even so, while some things can be put off until later, they can't be ignored; security needs to be considered from the outset and hardware might be the way to go if the product has safety critical aspects – interlocks, for example.

Image: Adobestock

In Greendale's opinion, the partitioning stage provides an ideal opportunity for design reuse. "If you have a design where you need processing, a power supply and a motor controller, for example, these separate elements give design flexibility. We could take a custom approach, but if the design can be created – even in part – using existing solutions, we'd be crazy not to use them."

Now comes the question of how to develop the hardware. "While getting everything on one board is good for production," he noted, "you can run into problems when starting a design from scratch. And, if you're laying out a large complex board, you wouldn't want to delay developing the power supply, for example. So these aspects need to proceed in parallel."

Power supplies are an increasingly challenging aspect of product design; not only does the design itself have particular requirements, there are also regulatory issues, including emissions. "In some instances," Greendale said, "we would rather develop the power supply on a separate board, allowing development to proceed in parallel and supporting reuse."

What's the volume?

And whether the product ends up as one board or several can be determined by target volumes. "If it's going to be made in the millions, that has to be considered from day one – manufacturability and testability are key. If it's an entirely new product, then you might need to demonstrate functionality first, then further partition the design so each element can be developed separately."

If it's a high value, low volume design, then yield becomes an issue. "If you then put a lot of complex functionality onto a single board," Greendale pointed out, "that may have a negative impact."

Target price is another consideration. "Often, the customer will have their own idea of a bill of materials (BoM) cost. If not, we can look at similar products to get a ball park figure. That's often more the case with high volume products; with low volumes, it's more about the amount of engineering that is needed. But if the client does have a target selling price, we can start from that and work out the percentage for the BoM."

But there are now many new entrants to the market who, rather than start from the traditional 'blank sheet of paper', will have started their design on something like a Raspberry Pi. "That's not a bad thing," Greendale accepted. "If that can prove an idea, then that's great – and the availability of peripherals can help with early development. But the problem we would then face is how to make the product smaller, less power hungry and lower cost. Those tasks are made easier if the product has already been demonstrated. And even if the engineering answer is 'start again', the 'Pi approach' will still have served a useful purpose."

"Tackling the problem is all to do with partitioning; deciding how best to split the solution into hardware and software elements."
Steve Greendale

Are there problems in moving code from the Raspberry Pi demo to, say, a product based on a Cortex-M0+? "Code can be more portable than it once was," Greendale asserted, "and when we develop a demonstration platform, it will be done with an eye to making the code portable."

But while the demonstrator might be Pi based, the right answer for a production version might be better suited to an FPGA, Greendale pointed out. "In that case, migration becomes a significant issue."

Optimising a design based on a Pi may be an issue. "When developing a new system," Greendale reflected, "it's often the things that people haven't worried about that prove to be the problem. So it's good to get the whole system together at an early stage; it allows you to develop code and to develop the functional blocks. Development doesn't have to wait until they're polished."

Another benefit of starting from something like a Pi is that a demonstrator could be more attractive. "People increasingly expect a demo to look like the final product," Greendale noted. "If you show them a concept, they expect it to be the right shape and size. Partly, this is due to the mobile phone industry packing a lot of functionality into a small product, but a small, developed platform can be a convincing start. Such an approach can also prove the algorithms work, but it's a powerful marketing tool – and that's an important element.

"A bespoke board can be a high cost solution, so proving an idea with a convincing demonstrator can be key to getting the money that's needed to develop the product," he concluded.