The relentless rise of embedded software means test and user centric interfaces are moving to centre stage. By **Chris Edwards**.

Software in the
spotlight

Embedded systems development is seeing a number of key changes as the software in devices becomes an increasingly fundamental part of their differentiation. In a roundtable discussion convened by *New Electronics*, experts pointed to how the rise in software is changing the approaches to development.

John Paliotta, CTO of test specialist Vector Software, explained: "Code bases are definitely getting larger. Teams are distributed now and life cycles are longer. When I first started, it was common that applications get recoded multiple times during their lifetime – even complicated systems. Now, teams and application sizes are so big you won't rebuild the code base.

"One of the other changes we see is that companies which historically outsourced development and testing are bringing development back in house and finding that software is their core competency."

More software development

John Carbone, vice president of marketing at Express Logic, pointed to the central role that software now has in many projects. "More and more of a project is devoted to software development. It's different to the case earlier, where there was a lot of hardware integration to be found. As MCUs have become more highly integrated, the hardware challenge has been simplified and standardised. Most designs today are based on ARM, which enables a lot of reuse from project to project."

According to Qt Company account manager Aurindam Jana, the user interface has become a cornerstone of many devices as the influence of the smartphone has spread. "Manufacturers want to see some

sort of iOS or Android like experience with the hardware they are working with."

Carbone agreed. "Definitely. The user interface has been an area of increasing functionality and focus, especially since the introduction of touchscreen interfaces like that on the iPhone. Apple recognised that people loved that interface. People now know how and expect to be able to pinch, zoom, flick and tap.

"These interfaces have now made their way into the Internet of Things, where devices are used to control the environment: home appliances and security systems. They now offer consumers the kind of interface they are familiar with and like. Companies are looking to develop and offer those kinds of interface much more than they did three or four years ago," Carbone added.

This can force a requirement to update the hardware to cope with better interfaces, said Jana. "What often happens is there is a product in the market a company wants to upgrade. It wants a new look and feel, but often these products were just sufficient to run the existing UI. If the new UI requires more computing power or memory than is available, the user experience will not match expectations. You would need then need to look at what

The need for interface portability is translating into frameworks that not only support today's platforms, but also tomorrow's

technologies can support the requirements."

Paliotta said: "Look at a phone. As a user, I don't look at which particular processor it uses. But my user experience? I care about that."

More portability

These changes play into development in two ways. One is an increasing focus on portability. If the price of an end device does not support the advanced interface functions of a multitouch interface, application developers need to contend with a simplified GUI. Jana explained: "You want to use the same core code base, but have different UI interactions. In Qt, something we really recommend is the MVC [model, view and controller] architecture. Here, the presentation layer, or UI, is different for each platform, but the model and control layers can be used across various systems."

At the opposite extreme are IoT devices that have no direct user interface. But even these can use the same MVC approach and the supporting frameworks. Jana said: "Qt is perceived as a GUI framework, but it has so many core classes that you could write a complete back-end and not have to worry about the GUI at all. The interface could be a remote client, talking over a protocol

> "When it comes to code quality, people have always been searching for the easy solution. But they don't always want to hear the prescription."
> **John Paliotta**

> "Customers are looking to spend less time putting development systems together and more on application development."
> **John Carbone**

> "Thinking back four years, would anyone have thought Android would be the main smartphone and tablet operating system?"
> **Aurindam Jana**

like HTTP or Bluetooth Low Energy to present data in a certain way."

MVC has a beneficial influence on testability and, with that, robustness. Paliotta said: "The way you design your software will greatly influence its testability. Every application that has a GUI should have a back end that does a lot of computation. The software now can have an abstraction layer between it and the GUI. Well designed software makes this layer as thin as possible. You are testing

what can't be automated. It's critical to design the software so the interface component is as thin and simple as possible, allowing it to be replaced in an automated test environment."

The requirement for interface portability is translating into frameworks that can support not just today's, but also tomorrow's, platforms.

"Can anybody guarantee what everyone will use in four years? Thinking back four years, would

anyone have thought Android would be the main smartphone and tablet operating system?" Jana asked. "They probably thought it would be iOS. When companies invest in their embedded software, how can they make sure their product is relevant in five year's time?

"It is an important focus for us. If a platform is of interest to us, we make sure we are on that platform. We will be on that platform and make sure that it works."

As devices become more connected, security and robustness are becoming bigger concerns, with ramifications for developers.

The requirement for platform portability drove the development of Express Logic's X-Ware Platform. Carbone said: "Customers are looking to spend less time putting development systems together and more on application development. Ever since the start of our company, we have provided what you could call example or demo projects. They work and they eliminate a lot of the the integration needed to get going. Now that systems have become more complex, that need has grown dramatically. It's hard to find complex systems with multiple components that can be used 'out of the box'. Typically, users will need TCP/IP, USB, filesystem and multiple components like that for whatever platform they are working on."

The problem is the hardware platforms provide generic drivers to support these functions and the assembled middleware may not work efficiently with those drivers.

"Two of them find out they have to access a certain resource and they bump into each other." Carbone said. "Developers have to do a certain amount of work to get started. So we have provided a platform that does that work for the user so they can start writing an application without having to worry about the detailed steps of integration. We've removed some of these integration gaps."

The growth in user-interface code, as well as overall project size, together with a shortage of developers, could point to an expansion in the number of languages used for development. But this is not the case, said Paliotta. "The situation with Ada was sort of unfortunate. I was a big fan, having used it and found the lack of integration problems that occur. But C and C++ have won. Python has also won as a scripting language. I have seen some C# and Java, but only at the periphery.

"We have a massive programmer shortage, but that shortage itself results in friction to change. A language would have to be remarkable to displace C/C++."

Using more than C

Jana said interest in improved interfaces is leading developers to use more than pure C. "We have seen renewed focus on C++. There has been big buzz about HTML5 for the UI. But, on low-level devices, customers are cautious about that particularly technology. They now prefer a native solution."

Carbone added: "We are definitely seeing a sprinkling of other languages, but it's mainly C. We are not seeing a substantial increase in languages other than C."

As devices become more connected, security and robustness are becoming bigger concerns, with ramifications for developers. Carbone said: "Where we find most of our customers are working is fairly deeply embedded into a product. The kernel we offer is small, mature, optimised and rock solid. We are used in billions of products – as far as reliability aspects are concerned, it's pretty solid. They don't need an RTOS to do anything different. What they do need are secure protocols, such as IPSec or SSL, plus gateway software to block unauthorised attempts to enter.

"Going outwards, as you add application software on top, that is where the focus on resilience needs to be – and that needs more rigorous testing. It's being exposed to other areas of entry. However, the RTOS can offer functionalities to make that easier, such as memory protection modules."

Paliotta said application writers need to look at changes in their methods. "Software code tends to be surprisingly undefensive. It's often written in a way that doesn't protect against out-of-range expectations. You need to look at robustness testing as well correctness.

"When it comes to code quality, people have always been searching for the easy solution. But they don't always want to hear the prescription. The way to lower your cholesterol is to eat less fattening foods. Test is only way to prove and document the correctness of software."

While languages will continue to dominate, as the code base continues to grow – not only in size, but also in importance – the issues surrounding embedded software, such as test and user-centric interfaces, will move centre stage.