The future of Embedded Systems

You Ain't Seen Nothing Yet

Wim Hendriksen

If you would have a choice, which time in the history of mankind do you want to live in? I would have selected today immediately. Listening to the Violin Concerto of Beethoven I write this story. One hundred years ago we had to hire a complete symphony orchestra to hear a performance of this violin concerto. That is a little bit outside the budget. What a pity that you could hear such a marvellous masterpiece only once in your lifetime. Or not at all. . .

Introduction

Last century human beings started to store and reproduce music. It was a struggle with a nail that scratched through the rills of a black plastic disk to reproduce music. We could hear the music through the rumble of the electrogramophone and we knew all too good what wow and flutter meant. When you turned up the bass and the volume, you could experience the concept of positive feedback with a loud booming sound.

Today, we can hear at home the world's finest performers - with a better quality than in the local concert hall - with our own stereo or home cinema system at any moment. Although music is just carefully moved air, you feel it is pure emotion. Your Quality of Life improves.

How is this all possible? By using Embedded Systems in our CD players, DVD players and Sound Processors. It is not difficult to find equal examples in cars, nuclear power plants, medical equipment, smart bombs, domotica or gaming devices: they cannot exist without embedded systems. Embedded Systems are invisible. You don't see them, you cannot drop them on your toes and they don't smell. Only when they don't work as expected, you notice their existence. They hide very modest in their embedding system. But what is an embedded system? In literature many definitions are found. In this paper we will use the definition of NetBSD, found in [1]:

> *An Embedded System is a combination of computer hardware and software and perhaps additional mechanical or other parts, designed to perform a dedicated function. In some cases, Embedded Systems are part of a larger system or product.*

Quality

One of the greatest misconceptions in the area of embedded systems is the insistence of developers to strive for zero defect products. Unfortunately, with today's means and methods this goal can be reached only in an infinite amount of time. Not one company has the financial resources to reach this goal. Of course for nuclear power plants you have different reliability requirements than for a MP3 player. But, even with all possible effort the zero defects goal still cannot be reached. So unfortunately this goal must be consigned to Utopia. It is better to accept a "just good enough" approach.

What this means is different for every product. For a pacemaker just good enough is that the pacemaker keeps the user alive during the lifetime of the product. So you need very high standards on reliability,

availability, robustness and lifetime. However the number of features on a pacemaker is limited and time to market is no issue.

We accept that a GSM phone breaks down after one or two years, and, when it locks up once a week, we know that we have to remove the battery for a few moments to make it work again. But the number of features increases year after year. Time to market makes the difference between a commercial success and commercial disaster.

What is the quality of a product? Is the quality of a Rolls Royce better than the quality of a Volkswagen Golf Diesel? No, car owners buy a product that fits their needs for a reasonable price. So value for money is an important issue, but also status, reliability, bells and whistles, image and design are important. Robert Pirsig, author of Zen and the Art of Motorcycle Maintenance [2] changed my mind by writing:

And what is good, Phaedrus,
And what is not good-
Need we ask anyone to tell you these things?

You know what quality is when you see it.

About three bugs per thousand lines of code can be found in commercial products with embedded software. And this number of lines increases exponential, so we have to live up with more and more bugs in the systems we buy. And still we buy these products. Why? Because customers do have different ideas about quality than the quality departments of embedded systems builders. Customers want the right product on at the right moment for the right price. So delivering a good product one month late is just as bad as delivering on a time a product with fewer features.

Here is an example:

Last month I bought a DVD player. After unpacking I read the owner's manual [3]. Somewhere in the middle of the book I saw the following instructions about a button on the remote control: "Press the P-scan button to activate the progressive scan feature for 480p output to a HDTV monitor. The unit powers down briefly and restarts in a new mode". I agree, this didn't look like a well thought out software architecture: I should have been warned.

On a separate inlay I read "Do NOT activate the Progressive Scan feature when the video output is set to the SCART setting in the setup menus; unstable operation of the RDV-1060 will result". So they ask me NOT to push a button on the remote control of a new piece of equipment. How long is an engineer able to resist such an order? I managed to avoid pushing the button for five long minutes. The DVD player died instantaneously when I finally did.

It took fifteen minutes before it worked again after following the "Troubleshooting instructions in case of No Power or unit freezes up after activating progressive scan in PAL systems." Maybe this piece of equipment should not yet have been shipped to customers.

But it looks good, sounds great, and displays a marvellous picture and all this for a reasonable price. So if I had to select again, would I have selected the same Rotel DVD player? Oh, yeaaah !

Challenges in Embedded Systems Design

Embedded systems are always designed in multidisciplinary teams. So people with backgrounds in electronics, mechanics, informatics or optics must work together to get the desired system. Nowadays projects are done in large to extreme large teams, sometimes divided over several sites, sometimes in different time zones, sometimes with different companies. Problems in optics are solved in software; problems in software are solved in electronics. Or vice versa, depending on the cheapest way, calculated over the whole lifecycle of the product.

For instance removing one of the power supplies from a product may save 500 Euro; with 3000 machines per year over 3 years this saves 4,5M Euro on cost of goods. You can program a lot of code for this amount of money, even though the software architecture may be not as transparent as before. (Because two motors in completely different subsystems are not allowed anymore to run simultaneously, the embedded software must perform some energy management).

First a team has to define the multidisciplinary requirements. Fortunately today processes and tooling are available to define requirements, but it only works in one discipline. So here comes the fun part: Mechanics, Physics and Electronics engineers are

used to work a little bit bottom up: "Let's start using known parts and see where we end". Informatics engineers in the meantime only want to think top down, so they refuse to think about solutions, they only want to think about requirements. In their hearts they want to start at the "oersoep," the beginning of life. After a while building this product starts with the architecture and the system design. As you can expect the bottom uppers are already far ahead, but are hindered by the top downers, who are asking nasty questions about the "what" and the "why" of the product. Things the first had forgotten to ask. With this way of working, you have a quick start, but not always in the good direction, followed by steering in the right direction. This may be the best of both worlds, only you don't learn it at school.

Somewhere halfway the project all disciplines meet again and in close cooperation the product is tested, debugged, verified and validated. When you look at such a project, it looks more like a jazz band than a symphony orchestra: everybody starts playing at the same time, everybody stops at the same time, but in between everybody plays his own solo without much attention to his peers. And still the audience can hear which song they play.

The story about top down and bottom up can be seen in a lot of embedded systems companies. Every discipline thinks that the others are complete and utter idiots, which often results in a lot of red faces, bad heartbeats, stonewalling and moaning: if everybody would use my methods, life would be much simpler. But the others have the same thoughts. Why don't we just accept that every discipline has its own design method, suited best for that discipline? So we are professionals in our own discipline. And in the meantime we must learn to communicate in the language of the other disciplines.

This is one of the reasons why formal methods never will survive in the embedded systems world: only people with informatics background are able to understand what is written down there, so nobody outside the own informatics group is able to review the output of these methods. As a result you very efficiently build the wrong system (twice), which was not the initial intention.

When you are able to understand what the restrictions of other disciplines are, then you are able to balance solutions in different disciplines. This means lots of communication in the project. Lots of communication is only possible when people are working close together. That is why projects designing mechanics in Eindhoven and software in Bangalore are doomed. But the managers don't see it until it is integration time. And then it's too late.

To probe further take a look at the Gaudí site of Gerrit Muller [4]. It gives a very good insight in the state of the art of embedded systems architecture, written by one of the most experienced embedded systems architects in the Netherlands.

Research

On a number of universities research is done in the area of Embedded Systems.

PROGRESS (PROGram for Research on Embedded Systems & Systems) wants to improve the knowledge in the area of Embedded Systems at Dutch universities and companies in order to improve the competitiveness of Dutch industry. PROGRESS has written an Embedded Systems Roadmap [5]. This book describes a vision of embedded systems and is used to steer research in the right direction. More information about PROGRESS and the running research subjects can be found on the PROGRESS website [6]. A new initiative of PROGRESS is to start with Public Outreach, which will disseminate the results of PROGRESS research to the appropriate people in companies, schools and universities. When you need more information about the results of PROGRESS research, please mail the author of this story.

ESI, the Embedded Systems Institute in Eindhoven, is organized around large Dutch embedded systems companies. Huge research projects have been started and will be started. Check the ESI website [7].

Dutch Institutes of Higher Education (HBO) have introduced so called "lectoraten". Lectorates about on e.g. embedded software, embedded systems and mechatronics are all up and running now. They mainly focus on pragmatic applied research and want to serve mainly the small and medium sized companies in the Netherlands.

In the European programs ITEA and MEDEA another set of projects is running, but it is difficult to

get an overview over of those these projects.

Education

In the future the Embedded Systems community needs people with a variety of skills on different levels.

For 30 years Institutes of Higher Education have delivered engineers on Bachelors level with knowledge of more than one discipline. A lot of the multidisciplinary architects today have this background. They are the pragmatic generalists who have found their way in the embedded systems jungle.

Unfortunately mathematics is taken out of the curriculum nowadays, while math is the Esperanto of the technicians. There is no doubt that in the future this will have its impact on the employability of these students in the embedded systems world.

Nowadays Institutes of Higher Education also deliver engineers with a Masters degree. These engineers are meant for the above mentioned excellent bachelors students. These Masters are more generalist or more specialist. An example is the Hogeschool of Arnhem en Nijmegen which delivers a masters study in Control Engineering. These Masters studies are also a good way to keep knowledge up-to-date of experienced engineers. They are also available in part time versions.

Dutch universities deliver Bachelors and Master with a more scientific approach: less generalist, more specialist. Universities are working on more multidisciplinary masters. The Universities of Eindhoven and Enschede are going to deliver Masters of Science for Embedded Systems students.

On the Stan Ackermans Institute of the University of Eindhoven you can get your MTD degree in Technical Informatics. I hope these people become the next generation of System Architects. An extra level of abstraction is needed in the complex systems of tomorrow and MTD's are able to cope with this. It is a pity that last year the Stan Ackermans Institute was broken up and organizationally placed under the ivory towers of the old faculties. This means that only mono-disciplinary MTD studies are possible from now on. A missed opportunity!

What we are missing is "post traumatic education". First you build a product in the real world and you have a feeling that you can do better. Therefore a fast track is needed for "born" embedded systems architect. Now it takes too long before they have enough experience to design large large-scale projects. Maybe a new future for OOTI?

Epilogue

We are building Embedded Systems for no more than 30 years now. And look where we are today. Now try to imagine what YOU can do in the next 30 years. The only restriction is your own creativity!

References

[1] Wim Smit, Wim Hendriksen (2003). *Embedded Systems, Smart and Intelligent Tools in an Increasingly interconnected globalized world* (ISBN 90-806440-2-1)

[2] Robert M. Pirsig (1974) *Zen and the Art of Motorcycle Maintenance* (ISBN 0-688-05230-4)

[3] Rotel Owner's manual RDV-1060 DVD audio/Video player 2003.

[4] Gerrit Muller: Gaudi site
http://www.extra.research.philips.com/natlab/sysarch/

[5] Embedded Systems Roadmap
http://www.stw.nl/progress/ESroadmap/index.html

[6] PROGRESS
http://www.stw.nl/progress

[7] ESI
http://www.embeddedsystems.nl

[8] Route67
http://www.route67.nl

About the author

Wim Hendriksen is part time Lector in the area of Embedded Systems at the Hogeschool van Arnhem en Nijmegen besides being an independent consultant in his own company Route67 [8]. He is involved in the applied research of multi- disciplinary requirements management and the application of embedded systems in the homes for senior citizens. Before 2000 he was manager software development at ASML and design engineer at ICT. He received his masters' degree in electronics at the University of Twente in 1979. He is a member of the advisory committee of OOTI and the program committee of PROGRESS. Email: wim.hendriksen@route67.nl or wim.hendriksen@han.nl