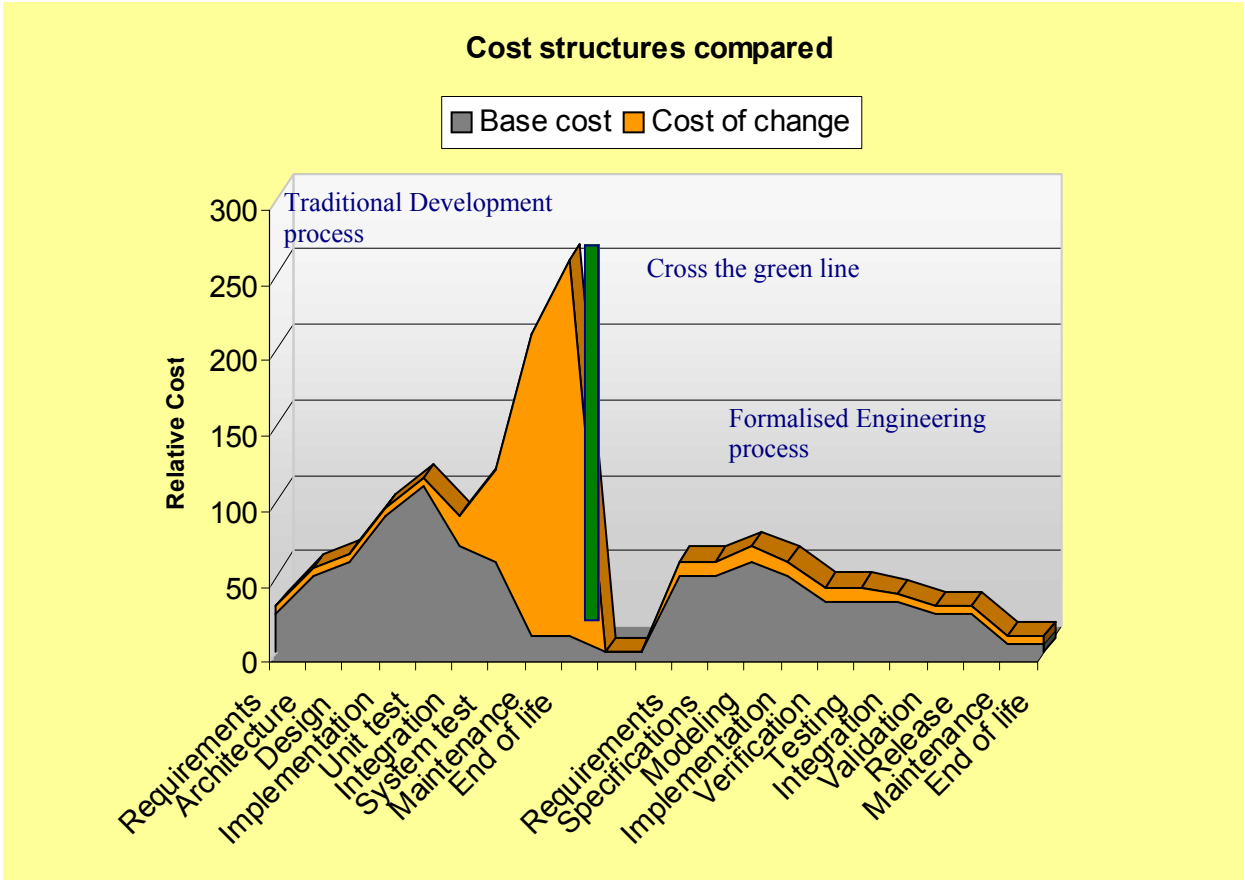
Time to Quality gives More for Less

with Altreonic's formalised systems and software engineering methodology and tools

Think before you begin. Some people might call this wisdom, but it is at the core of systems and software engineering. It serves two main goals:

Firstly, it serves to **analyse and understand**. What is the goal to be achieved? What is the real problem to be solved? Is it a real issue or does it just look like one? This analysis activity is often dominated by the use of natural language and by the presence of many stakeholders. One must be aware that at this stage the information will be incomplete, incoherent and contradictory, or will already assume a solution based on past experience. Making sure that all stakeholders agree on the product are system to be developed is called "Requirements and Specifications" capturing. At this stage, one must not only think in terms of the "normal" use, but also take into account fault and test conditions. If not considered up front, it can be very costly to retrofit a system with support for it.

Secondly, it serves **to plan and to predict**. Once the requirements and specifications are well understood, architectural modelling can explore different ways to implement the specifications. Simulation modelling allows "what-if" analysis and allows to verify that the requirements and specifications really reflect what is needed. Formal models can then be use to verify mathematically critical properties of the system.
Once all alternatives have been considered, the implementation can start and we enter the domain of more traditional development engineering. The architectural modelling will have identified several entities that provide the specified properties. These can be assigned to a Work plan that divides the work into Work Packages around a clustered set of entities.



info.request@Altreonic.com
www.Altreonic.com

Altreonic NV
Gemeentestraat 61A b1
B3210 Linden, Belgium
Tel. +32 16 202059

Work plan view

A **Development Task is** the core activity of a Work Package. It will develop a part of the final system according to a predefined or mandatory methodology. To verify that the development was done correctly a **Verification Task** is executed and when no issues are found, **Test tasks** will verify that the developed entity meets the specifications.
When all development is done, verified, and tested integration can start. When completed, the system can be validated against the original requirements and a release can be defined. If all goes well, the selected architectural model will be the implementation model and a release can be defined. This transfers the developed product or system to production entering the phase of maintenance.

More upfront, less costs afterwards

In general, such an engineering process is not a linear but an iterative one. During the process, issues will be discovered and changes will be needed. But the further the project has proceeded, the higher the cost will be if an issue results in rework. Once in production, it can even cripple companies. The keyword in all these activities is formalisation. Often, it will even result in cleaner architectures
Hence, a well thought out engineering process will not only result in better quality for less cost, but also provides risk reduction by finding the issues as much as possible early in the process.

Altreonic provides more

At Altreonic we not only formalised a specific engineering process, we took a formalised look at systems engineering in general and the resulting view is surprisingly simple. We found it can be used as for technical as well as for non-technical domains. It can even be applied in a project to develop a standards aware methodology. To support such an engineering process with the required level of productivity we developed tools as well.

OpenCookbook

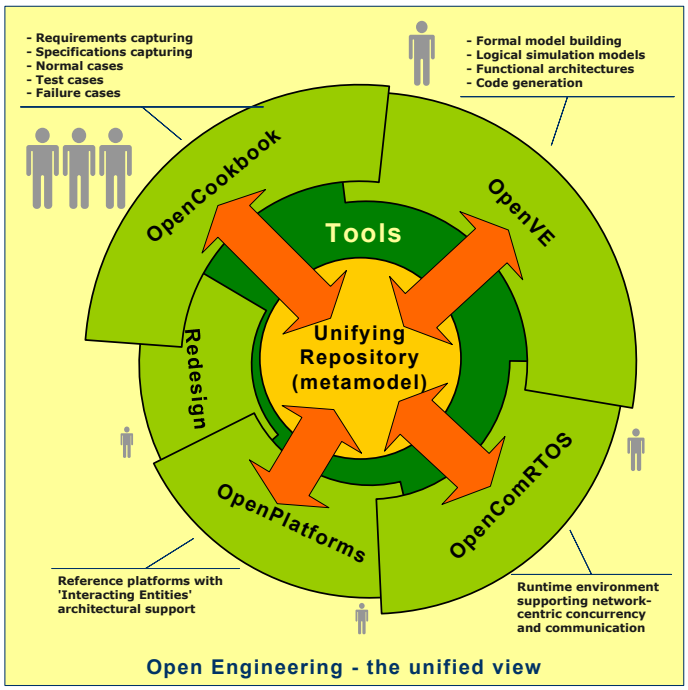
OpenCookbook was developed as a web based environment supporting a project from requirements till reaching the release point. However in an organisation heuristic knowledge is often the key in preserving the competitive edge and such knowledge and organisational procedures can be captured as well. If compliance with external standards (like the IEC61508 safety standard) is needed, these can be entered as boundary conditions to the engineering process allowing to pre-certify while the project is executed. By using a web based approach, each project becomes a real-time portal, to be accessed and used easily by a distributed team. Documents are not written but generated as time stamped snapshots.

OpenVE and OpenComRTOS

As many real-world systems can easily be modelled as a set of interacting entities, this became our main architectural paradigm. This is reflected in OpenVE a visual modelling environment. Being generic, it can support different RTOS but in particular OpenComRTOS. The latter was formally developed as a network-centric RTOS allowing to program from very small processors to heterogeneous distributed networks in a transparent and scalable way. The use of formal modelling has resulted in a very clean and safe architecture within a very small code size, ranging from just 1 KBytes to about 10 KBytes. Small in the embedded world means more performance and less power.
Using OpenVE and OpenComRTOS, the user can even define his own services, simulate his distributed application on a Windows or Linux PC or even include the latter as fully functional nodes in his systems. Remapping to a different topology is a matter of seconds and mainly requires recompilation.

Lessons

What Altreonic brings is less risk and higher quality at a lower price by shifting the effort upfront and by providing an integrated approach from early requirements to product release. In these times where trustworthiness is important, this means not only less time to market but also less time to quality. And quality in the end is always a winner.



Open Engineering - the unified view

For information or to discuss your needs, contact us at:

info.request@altreonic.com
www.altreonic.com