Managing the evolution of a system is becoming increasingly critical to successful development in the embedded software industry. Those systems are no longer closed and standalone. Instead, they often interact with all kinds of devices in a networked environment, where each application might need slightly different software versions. Additional user requirements, changes of the underlying hardware, as well as emerging new system and software architectures necessitate ongoing software updates. It should be possible to keep track of the changing requirements and to design all these versions in a uniform way. At the same time software quality and certification has to be closely monitored. Finally, systems will need to adapt automatically and/or allow – remote – updates of components, often without interruption to the service. EMPRESS addresses these challenges systematically with the RUP-based EMPRESS process.

The EMPRESS approach
In EMPRESS we have developed methods, tools, and a process for real-time embedded software development that supports evolution management in a flexible and dynamic way. The two major cornerstones of evolution management within EMPRESS are an adaptable component architecture and an incremental requirements management and validation approach. Both have a solid formal base for defining evolution paths. They are coalesced by the EMPRESS process.

The benefit of this approach is a rapid and more efficient way of developing and/or adapting high quality embedded systems (versions/product lines). During run time less effort is required when it comes to updates.

EMPRESS proposes component-based development. Components can evolve independently. However, they may influence others as well as the overall component architecture. To exploit the advantages of components to the fullest extent to the benefit of evolution management, components are included in the earliest design stages, with a special emphasis on component interaction.

A component-based architecture for evolution management
An adaptable component-based architecture and a stable enabling infrastructure have been developed that provide support for evolution during design-time (development) and at run-time.

Using architecture-pattern-based analysis and built-in testing, it can be assured that a component meets its functional and non-functional requirements. Flexible (parameterisable) components, layering, interface wrappers, and dynamic configuration enable adaptation of components to their target environment.

Major project outcomes

**Dissemination**
- 26 papers (including conference presentations)
- two workshops
- two journal articles
- one book chapter

**Exploitation**
- two contributions to commercial tools
- two new external service
- three new services for internal purposes
- many improved internal and external services
To bring a component into operation without interruption of the system, communication patterns (e.g., the bridge pattern) and state description techniques are defined. Resource aware systems take care of system constraints, such as timing.

Incremental requirements management and validation
With respect to requirements engineering and management, EMPRESS’ methods and techniques serve to form a complete and consistent set of requirements as a base for later evolution, and to maintain the set of requirements: i.e., update it to the needs of the evolving system, track them through the development process and provide data to assess the impact of coming changes. They cover all requirements engineering and management tasks. However, they all have special emphasis on incremental system changes. The methods include elicitation and documentation of non-functional requirements, classification schemes for safety critical systems, tracing of requirements, estimation of impact of change, management of configuration and change management.

The validation and verification focuses on the incremental approach. Built-in integration testing eases the validation of a new component’s behaviour. New validation (sub-) processes have been developed specifying incremental strategies for module integration and hardware/software integration. Based on these strategies initial ideas for a framework for certification of component-based systems has been formulated.

A process enabling the development of evolutionary systems
The EMPRESS process is based on the Rational Unified Process (RUP). The major additions are:
• new or enhanced methods to deal with evolution; the focus is on the disciplines, which are most affected by evolution (see figure),
• the extension of the test to a verification and validation discipline to emphasise the importance of software quality,
• a new phase run-time at the end to address the issues of runtime updates.

EMPRESS is a process framework for evolutionary embedded real-time software development. It provides support for all development phases, with a focus on the early ones, but also extends to run-time. It contains methods, techniques and tools to cope with evolution, and it shows when, and for what purpose, they should be applied. As a framework it serves as the basis for a project’s process by tailoring the framework to a company’s and/or a project’s needs and adding missing parts.

Major achievements for real-time embedded software engineering
EMPRESS has achieved an impressive contribution to standards, methodology, process and tools for real-time embedded software engineering. The project results will yield the following benefits:
• industry partners will use them to improve their development process, reduce time to market, and increase global competitiveness in many application domains;
• the methodologies will be offered as consultancy services and products so that the entire European real-time embedded software industry benefits;
• as soon as they are sufficiently mature, the advanced tools developed in EMPRESS will be available from major software providers.