# Architectural Framework for Energy Optimization in Embedded Systems

**Hasan Sözer, Arjan De Roo, Mehmet Akşit**

Software Engineering Group,
University of Twente,
Enschede, The Netherlands
{sozerh, roo, aksit}@ewi.utwente.nl

## Abstract

*Energy consumption has become one of the important system properties that should be controlled by embedded control software. There is usually an inherent trade-off between energy consumption and several system qualities. As such, optimization techniques should be adopted for making the desired trade-off among quality attributes. Implementations of these techniques are usually ad-hoc and system-specific, leading to implicit design decisions distributed over several software components. We propose an architectural framework for custom synthesis of control software from reusable and programmable elements. The goal is to facilitate systematic reuse of knowledge in the optimization domain and explicit management of quality trade-offs for energy optimization of embedded systems.*

## Keywords

Energy consumption, embedded systems, optimization

## INTRODUCTION

Software has become an integral part of today's embedded systems. For example, overall system operations and properties in digital document printing systems are mainly controlled by software. Energy consumption has recently become an important issue and as such, it has become as one of the important system properties that should be controlled by the embedded software. However, there is usually an inherent trade-off between energy consumption and several system qualities. For instance, customers demand minimal energy consumption for printing systems. However, they also demand high productivity, which requires running the machine at a higher speed, and which in turn increases the energy consumption. Moreover, the print quality should also be kept at acceptable levels, which requires running the machine at a lower speed. Therefore, embedded control software must take multiple, conflicting objectives into account and choose the optimal values for control parameters (decision variables), while keeping energy consumption below the allowed limits. Examples of control parameters are the power given to a component and the speed of the system. In addition, the software should employ adaptive control to retain optimal system qualities under varying circumstances such as changing context/environment, usage profile and evolution of system properties. The constraints for control parameters should also be taken into account. For example, a component needs a certain amount of power under a given speed. In essence, this is an optimization problem known as multi-objective optimization (MOO) [1]. There exists a vast literature on quality measurement/estimation, trade-off and optimization techniques. However, research efforts mainly focus on mathematical problems in this domain rather than the application of the devised solutions in real software systems. The realization of these solutions for embedded systems is usually done at the implementation phase of the software. The solution is distributed over several components, and possibly implemented by different engineers. This ad-hoc approach results in several custom implementations and implicit design decisions. However, the design decisions regarding the quality trade-offs are critical for an embedded system and they should be made explicit at the software architecture design [2]. In addition, there is a well-established theory and a set of solution techniques for MOO, which can be systematically reused. Therefore, we propose a flexible, architectural solution in terms of a framework that comprises reusable solutions for energy optimization in embedded systems. The framework aims at facilitating *i)* systematic reuse of knowledge in the optimization domain, *ii)* explicit management of quality trade-offs with respect to the energy consumption, and *iii)* custom synthesis of MOO solutions from reusable and programmable artifacts. In the remainder of the paper, we discuss our solution approach and related research directions.

## THE APPROACH

Our approach is presented in Figure 1. The architectural framework consists of three main elements *i)* reusable artifacts that can be tuned and composed together with tool support, *ii)* an architecture synthesis method which guides the software engineer for utilizing these artifacts, and *iii)* a set of analysis tools, which provide feedback with respect to the quality attributes of the design instantiated for a particular application. The analysis process takes a set of provided quality requirements and constraints into account. As long as these requirements and constraints are not met, the software engineer makes iterations to improve the design. Once the design is complete, (partial) code generation is possible since the design elements are based on predefined reusable artifacts.

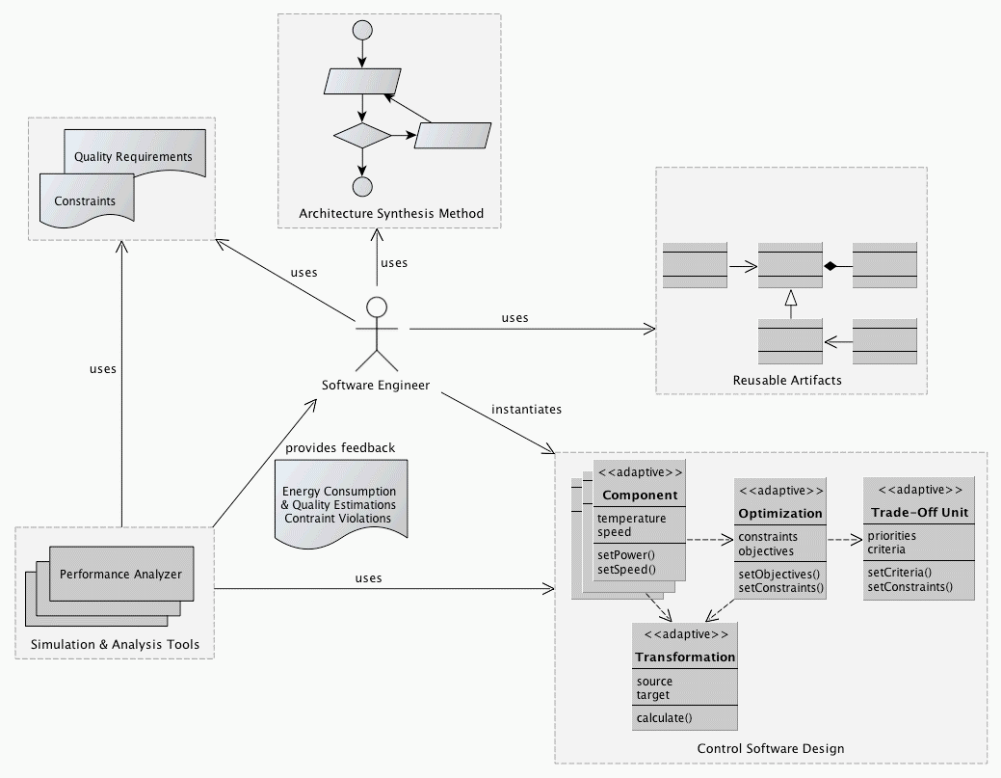


**Figure 1. The Overall Approach**

As the basic requirements, the framework should provide the following:

- Support for the formal specification of quality requirements and constraints including the ones related to energy consumption.
- Architectural styles and a uniform notation that enables the instantiation and specification of software architectures.
- As a part of the notation, support for expressing energy and quality related properties/semantics used for automatic generation of formal models as inputs for simulation and analysis tools.
- Extensible artifacts enabling the incorporation of additional optimization and trade-off techniques, when needed.

## FURTHER RESEARCH DIRECTIONS

Each of the basic requirements, as indicated above, can be extended with sub-questions. For instance, the notation for specifying the architecture can support multiple views. Some of these views would require special annotations and/or elements for energy consumption. Similarly, formal models and analysis techniques needed to be extended for the estimation of energy consumption. Another issue, which is implicit in the current approach, is the integration of the developed control software with the system.

## ACKNOWLEDGMENTS

The ideas presented in this paper are based on our work that has been carried out as a part of the OCTOPUS project [3]. This project is managed by the Embedded Systems Institute and it is partially supported by the Netherlands Ministry of Economic Affairs.

## REFERENCES

[1] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, Wiley, UK, 2001.

[2] A. de Roo, H. Sozer and M. Aksit, “An architectural style for optimizing system qualities in adaptive embedded systems using Multi-Objective Optimization”, In Proceedings of the 8th Working IEEE/IFIP Conference on Software Architecture, pages 349-352, Cambridge, UK, 2009.

[3] Octopus project, Embedded Systems Institute, [online] http://www.esi.nl/projects/octopus, Eindhoven, The Netherlands, 2010.