Hardware/Software Partitioning Methodology for Embedded Applications using Multiple Criteria Decision Analysis

Gaetana Sapienza

2016

School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden
To everyone who supported me during this journey
Abstract

New hardware technologies enable execution of embedded systems applications on heterogeneous platforms. These platforms consist of different execution processing units, for example CPUs, and FPGAs, that enable the application execution of software (SW) components, typically implemented as C/C++ code, and hardware (HW) components, implemented, for example, as VHDL code. This heterogeneity enables building dedicated components which can significantly improve application performance. However, it also requires decisions about which components will be implemented as SW and which as HW execution units. Such a decision process is known as HW/SW partitioning, and has been subject of research for more than 20 years. A typical goal of research has been to find the optimal partitioning with respect to system performance, and possibly a number of other properties such as power consumption, or resource utilization (e.g. related to CPU memory footprint and FPGA area). However, due to the significant increase in complexity of the applications, and inclusion of different requirements, partitioning decisions have become more complex, has the entire development process with an integrated partitioning decision process. Today there is a lack of a systematic approach for partitioning complex applications, and this thesis addresses this challenge. In particular, the main objective of the thesis is to design and build a systematic partitioning decision process that considers many requirements of different types. The thesis describes a new method, MULTI-PAR, that includes the partitioning decision process for component-based embedded systems. The method is based on model-based-engineering principles; components are analyzed as models which can be implemented either as a SW or HW components, and the implementation itself is performed at a late stage of the development process. The partitioning is based on the optimization of the application’s and components’ extra-functional properties (EFPs) that are derived from the requirements and project constraints. For the optimization a Multiple Criteria Decision Analysis (MCDA) method is used. As part of the main contribution, the thesis includes several independent contributions that are of a more gene-
rational character: a) modeling principles for component-based applications which consist of SW and HW components, and components can be implemented as SW and/or HW code; b) a classification and analysis of EFPs in respect to the dependency on their HW or SW implementation; c) composition rules for some of the EFPs of SW and HW components; and d) suitability analysis of MCDA methods in their usage for the partitioning decisions. MULTIPAR is also implemented in the form of a tool that enables a selection of components and analysis of the system in respect to the selected EFPs. The feasibility of MULTIPAR was validated through two industrial cases. The thesis is organized in two parts; the first part includes an introduction summarizing the overall work and discussing the research approach, and the second part collects the most relevant papers published in different venues.
Sammanfattning

varukomponenter; d) Lämplighet och begränsningar vid användande av multikriterianalys för partitionering. MULTIPAR har också implementerats i ett verktyg och har validerats i två industriella fall. Avhandlingen är indelad i två delar. Den första delen summerar på ett övergripande sätt arbetet och forskningsmetodiken. Den andra delen består av det för arbetet mest relevanta publikationerna.
Acknowledgements

I would like to express my deepest gratitude to my main advisor Professor Ivica Crnkovic for his valuable guidance, and for “always being there” during this journey.

I am very thankful to my industrial mentor Professor Tiberiu Seceleanu for the valuable guidance during these years.

I also would like to thank Peter Löfgren, Helena Malmqvist, Johan Åkerberg and Krister Landernäs for their constant support and encouragement.

My thanks are extended to my colleagues and managers at ABB Corporate Research, and colleagues and professors at Mälardalen University.

Lastly, a special thanks goes to my family.

This research is supported by ABB Corporate Research, the Knowledge Foundation through ITS-EASY, an Industrial Research School in Embedded Software and Systems, and by the Swedish Foundation for Strategic Research through the RALF3 project.

Gaetana Sapienza
Västerås, Sep 2016
List of Publications

Publications


Related Publications

These publications are selected among the other not included publications, since they have contributed to the thesis results.

Thesis


Journals and Magazines


Conferences and Workshops


Technical Reports


# Contents

I  **Thesis**  

1  **Introduction**  

2  **Research Overview**  

2.1  Research Objective and Research Questions  

2.2  Contribution  

2.2.1  Contribution Context  

2.2.2  Main Contribution  

2.2.3  Contribution through the Included Papers  

2.3  Research Approach and Validation  

2.3.1  Research Methods  

2.3.2  Validation  

3  **Related Work**  

4  **Conclusion**  

4.1  Summary  

4.2  Remaining challenges  

4.3  Answers to the Research Questions  

4.4  Future Work  

Bibliography  

II  **Included Papers**  

5  **Paper I:**  

---  

---  

---
Contents

5 Paper I: Multiple Properties 45
5.1 Introduction ........................................... 47
5.2 The Metamodel for Partitioning ...................... 48
5.3 The MultiPar Process .................................. 50
5.4 Wind Turbine: Industrial Case Study ................. 53
5.5 Related Work ........................................... 57
5.6 Conclusions ............................................. 57
References .................................................. 58

6 Paper II: Architectural Decisions for HW/SW Partitioning Based on Multiple Extra-Functional Properties 61
6.1 Introduction ............................................. 63
6.2 Modeling HW/SW component-based systems ........... 65
6.3 The partitioning quality framework .................... 68
   6.3.1 Component property metamodel ................... 68
   6.3.2 The categorization of partitioning-related properties . 70
6.4 Survey of EFPs in the Automation and Control Domain ... 72
   6.4.1 Objective ............................................ 74
   6.4.2 Survey Design and Process ......................... 74
   6.4.3 Results .............................................. 76
   6.4.4 Analysis .............................................. 76
   6.4.5 Validity .............................................. 78
6.5 Related Work ............................................ 82
6.6 Conclusions and future work ........................... 84
References .................................................. 85

7 Paper III: On Applying Multiple Criteria Decision Analysis in Embedded Systems Design 91
7.1 Introduction ............................................. 93
7.2 Research Context and Related Work ..................... 95
   7.2.1 Basic Concepts of Multiple Criteria Decision Analysis 95
   7.2.2 Partitioning and MCDA Suitability Analysis Related Work .... 100
7.3 MCDA Methods - Partitioning Suitability Criteria .......... 102
7.4 Assessment of MCDA Suitability for Partitioning ............ 106
   7.4.1 MCDA Methods vs. the 11-Suitability Criteria .......... 106
   7.4.2 Suitability Assessment ............................. 110
9.3.4 Result Discussion .............................................. 198
9.4 Conclusion and Future Work ................................. 202
9.5 Acknowledgments .............................................. 203
References .......................................................... 203
9.6 Appendix: PDT - MultiPar Tool .............................. 207
9.7 Appendix: Integrating Evidential Reasoning into MULTI-
PAR .............................................................. 211

10 Paper VI:
Inclusion of Ethical Aspects in Multi-Criteria Decision Analysis 215
10.1 Introduction ...................................................... 217
10.2 Background .................................................... 218
10.2.1 Multiple Criteria Decision Analysis (MCDA) .......... 218
10.2.2 Values and subjectivity in the decision process and MCDA . 220
10.3 The importance of Ethical Aspects ......................... 220
10.3.1 Consequences of neglecting ethical aspects in decision
making process .................................................... 221
10.3.2 Ethics in technology and other fields with high impact .... 222
10.3.3 MCDA and Ethics ......................................... 223
10.4 Explicating ethical aspects in MCDA ...................... 223
10.5 Case study: HW/SW partitioning for a wind turbine APPLI-
CATION .......................................................... 225
10.5.1 The partitioning problem .................................. 225
10.5.2 Ethical aspects of HW/SW partitioning ................. 226
10.5.3 Case study: HW/SW partitioning ....................... 227
10.6 Conclusions and Future Work ............................ 229
References .......................................................... 231
I

Thesis
Chapter 1

Introduction

Heterogeneous platforms, i.e. platforms consisting of different computation resources, e.g. Central Processing Units (CPUs), and Field-Programmable Gate Arrays (FPGAs), allow the deployment of embedded applications as hardware (e.g. synthesized Very High Speed Integrated Circuit Hardware Description Language (VHDL) code), and software (e.g. compiled C/C++ code executable units). A heterogeneous platform makes it possible to take specific advantages of hardware and software execution units, for example hardware (HW) parallel execution and software (SW) design flexibility [1].

At design time, it is of crucial importance to carry out architectural decisions on which parts of the application will be deployed as HW units and which parts as SW units, because the deployment “has a first order impact on the cost/performance characteristics of the final design” [2]. The separation of an embedded application into HW and SW parts (or components, or execution units) is known as the “HW/SW partitioning”. This is an old concern, which has been addressed in the literature (see for example [2–4]) over the last two decades. It challenges designers to find an optimal combination of HW and SW components in order to reach the application development goals. Most of these goals can be expressed via extra-functional properties (EFPs), which are properties describing the quality characteristics of a system or component. Examples of EFPs are performance, energy consumption, time-to-market, and safety [2, 5]. Many of these EFPs depend on HW/SW partitioning decisions, i.e. whether a component will be executed as a SW or HW component. As the application complexity increases, along with the number of EFPs considered in the design process, achieving the optimal HW/SW partitioning decisions
becomes more challenging.

In industry a widely used practice is to carry out partitioning decisions in an early stage of the design phase. After this, the detailed design and implementation of HW and SW components continue as separated parallel processes. The integration between HW and SW is performed at a late stage of the implementation phase. At integration time, due to the fact that the HW and SW designs have evolved independently, redesigns are often needed. Such an approach will likely have a negative impact on overall development time and final application quality. Moreover, the partitioning decisions performed in the beginning of the design are often driven by implementation platform specifics, which make it difficult to reach goals such as portability, interoperability, and reusability [6].

The partitioning state-of-the-art shows that decisions are usually based on a limited set of EFPs, which are mostly related to the runtime behavior of the application such as performance, energy consumption, memory footprint and FPGA area (e.g. [7–9]). The EFPs related to lifecycle aspects, project management and business goals are not taken into account when performing partitioning decisions.

Focusing on only a limited set of runtime EFPs might lead to longer development process and negatively affect the system lifecycle. For instance by neglecting a property such as maintainability, it might be hard to support field-upgradeability of those functions which have been implemented in VHDL instead of C code. Another simple example can be given by considering the design effort involved: the design of a HW function might require more effort than that needed to reuse an existing SW function, so it will impact the overall development process in terms of cost and time. In general, there is no support for analyzing relevant properties in the context of partitioning such as system cost, flexibility, system design time, maintainability, testing cost, portability, or reusability. Moreover, taking into account more EFPs will require addressing more conflicting decisions in relation to the number of EFPs under analysis. Designers will be challenged even more when performing the partitioning decisions, because they have to cope with a large number of factors and possible interdependencies. For instance, performing decisions to decrease the execution time might lead to a situation in which the energy consumption increases requiring additional design effort. Including a large number of EFPs in the partitioning decision process increases the need for a way to methodologically enable confronting trade-offs between the EFPs, and supports engineers in specifying and comparing the different types of EFPs. In addition, in order to minimize unnecessary redesigns and reach application development goals, it is important to push the platform-specific design into a late stage of the design
phase. This requires a solution able to provide a partitioning decision approach that facilitates technology independence in the initial stage of the design and pushes the technology-dependent design to a late stage.

This thesis addresses the challenges and needs mentioned so far. In particular, our main objective is the design of a process which is able to support engineers in making partitioning decisions which consider a large number of EFPs of different kinds: runtime, lifecycle, project management and business goal-related EFPs. Our solution combines a) multiple-criteria decision analysis (MCDA); b) component-based development (CBD); and c) model-based design (MBD). MCDA is a sub-discipline of operational research that has been widely applied in many fields such as, for instance, economics, medicine, energy management and manufacturing to take decisions based upon several criteria from a stakeholders’ perspective [10]. It provides structured and transparent methods to break-down complex decision problems. CBD is a well-established approach in software engineering that provides architectural solutions, interaction standards, and techniques to reuse application components [11]. We apply CBD to exploit reusability and enable formal definition of the components deployable as HW or SW execution units. MBD provides techniques to enable the separation of the design into platform-dependent and -independent stages [6]. MBD allows technology-independent design to be performed at the beginning of the design stage and platform-dependent partitioning decisions to be postponed until a later design stage.

To reach the intended goal we designed MULTIPAR, a methodology that provides a partitioning decision process for component-based embedded applications. The goal of MULTIPAR is the optimization of the partitioning (i.e. HW/SW configuration) with respect to the selected EFPs. By analyzing the application development goals the EFPs of interest for the partitioning of a given application are identified. The components that constitute the application architecture are represented as models. Every component can be implemented as several variants, including HW and SW variants. Each variant is described through its interface and a set of EFP values. Reusable variants are stored in a repository. The optimization is performed in two steps, one which considers the available component variants, and the other one which considers their combinations. In particular, the first step focuses on the selection of the component HW and SW variants that optimally satisfy the EFPs of interest for partitioning at component level, while in the second step, the optimization of all the possible system variants is carried out. Both optimizations are performed by using MCDA methods. MCDA methods are able to compare qualitative and quantitative EFPs of different kinds, such as runtime, lifecycle, project-management
As a part of the main contribution, the thesis includes: a HW/SW partitioning decision process as a part of the overall system design process; a component model that includes specification of both HW and SW components; a classification of EFPs of interest for partitioning, in which the EFPs are divided into three classes: lifecycle, runtime and project/business related, as well as an analysis of their impact with respect to the partitioning in the automation domain; a set of composition rules for deriving application EFPs from component EFPs; a suitability analysis of MCDA methods for the partitioning processes. The proposed methodology is implemented as a software tool and has been validated from a feasibility perspective through two industrial cases related to the automation domain.

The thesis is organized in two parts:

**Part I** describes the research problems, the related research questions and the research approach. Further, it gives an overview of scientific contribution and presents the related work. It ends with a section discussing the achieved results and a section which concludes the research work and indicates possible future research directions.

- Chapter 2 presents the main research objectives and related research questions, the main contribution, the undertaken research methodology, and finally the validation of the research work.

- Chapter 3 discusses related work.

- Chapter 4 concludes the presented research work and proposes future research directions.

**Part II** includes six papers that comprise the research work.
Chapter 2

Research Overview

This chapter presents the main research objective and the related research questions. It summarizes the research contribution presented in the publications included in the thesis. It presents the research approach and the validation performed.

2.1 Research Objective and Research Questions

The main research question posed in our work is:

How to provide support for HW/SW partitioning of embedded applications that, in a systematic way, considers all important requirements and constraints to reach the application development goals?

Due to the increased complexity of highly demanding performance embedded applications, market demands and technology trends (here specifically in the form of heterogeneous platforms), HW/SW partitioning decisions are today dependent on many requirements and constraints. These requirements and constraints are not only related to runtime properties such as performance, memory/area utilization, power consumption, but are also related to lifecycle properties (e.g. safety, maintainability, sustainability), project management constraints such as available expertise, intellectual property costs, and legislation, and business goals such as development of mass-products or product-line, and localization. Trading-off all of these requirements and constraints for find-
ing the most suitable partitioning solution significantly increases the complexity of the design process, in particular in relation to the HW/SW partitioning decisions of the embedded application. For instance, if a component requires a high-parallel computation it might be more suitable to implement it as a HW component to execute on FPGA, but in the case of meeting a requirement such as the upgradeability, it might be more suitable to implement it as a SW component and run it on a CPU.

Our goal is to attain the design of a methodology for enabling the partitioning of embedded applications into heterogeneous platforms, taking into account many and different requirements and constraints related to runtime, lifecycle, project management and business goals.

We started our research by identifying the key steps and artifacts that allow the establishment of a systematic procedure for carrying out HW/SW partitioning decisions. This led us to the following question:

**RQ1. What would be a systematic decision process to support engineers in performing HW/SW partitioning of embedded applications?**

Answering RQ1 requires the definition of a HW/SW decision process as part of the entire development process. In this HW/SW decision process the decision criteria representing the requirements and constraints of interest are expressed in form of EFPs. However, not all EFPs are of relevance in making HW/SW partitioning decisions. Consequently, we envisage the necessity of identifying which EFPs are influenced by the HW/SW partitioning decisions, as addressed by the following question:

**RQ2. Which EFPs depend on the HW/SW partitioning and in which way?**

In order to achieve an optimal (or nearly optimal) partitioning with respect to all relevant EFPs, the EFPs need to be compared, which is challenging. The EFPs can be numerous and expressed by using different metrics. Thus, we aim to identify the proper methods and related tools for handling the property diversity which allow a large number of EFPs to be taken into consideration. Based on this, we formulate the last question:

**RQ3. What methods and related tools are suitable for specifying and comparing the EFPs in the HW/SW partitioning process?**
2.2 Contribution

This section is divided into two parts. The first part introduces a few key concepts on which our work is built. The second part highlights our main contribution.

2.2.1 Contribution Context

The embedded applications we are considering are component-based applications. CBD techniques increase development efficiency and minimize development costs by facilitating the reuse of pre-existing components. CBD techniques are well-established in software engineering. They can also be applied to the development of embedded applications where components can be implemented as SW and HW executable units. In order to further benefit from the reusability of existing components and increase design efficiency, we combined CBD techniques with MBD techniques. The latter have the benefit of enabling platform-independent design and providing support for the design of industrial applications for control, signal processing and communication systems. By adopting such a platform-independent approach in the early stage of the design, the application can be architected as a number of interconnected components abstracted from their implementations. Later, these components can be implemented either as HW components (i.e. VHDL-code) or SW components (e.g. C/C++ code). Both HW and SW component implementations (here also referred to as variants) can already exist and consequently be reused from a repository of existing components. They might also not be available and need to be designed from scratch. In this case they will enrich the existing repository for reuse in future development process.

Different variants of the same component implement the same functionality and expose the same interface. However, EFPs are in principle different for different variants. For instance, in case of a component which has a SW and a HW variant, both variants implement the same functionality, but their runtime behavior might be different; for example the SW variant might have worst-case execution time (WCET) greater than the HW variant.

Traditionally, the design phase starts with the analysis of the overall requirements, which aims to identify what functions are needed. Our focus is on the partitioning of the application into HW and SW executable components, given the application architecture. The latter is designed as a number of interconnected component models. We assumed that the application architecture with the identified component exists. The architectural design itself is not in
the scope of our research but rather our method starts with the identification of the EFPs on the system and component level. The EFPs are derived from the requirements, constraints and goals. At component level the EFP values are related to the component when it is working in isolation. At system level, the EFPs are related properties that are visible at the system and application level (this type of EFP is referred to here as a system EFP).

In our work we aim to provide support for performing partitioning decisions based on many, multi-faceted and even conflicting requirements and constraints. With this target in mind, we oriented our study towards a sub-discipline of Operation Research that has been widely applied in many fields such as economics, medicine, energy management, manufacturing, etc. to take decisions based upon several criteria and from several stakeholders’ perspectives. This is known as Multiple Criteria Design Analysis (MCDA) or Multiple Attribute Decision Making (MADM). The main advantages it offers are to allow many criteria to be dealt with (which in our case correspond to both components and system EFPs), facilitate our requirement of reusing existing components, and enable a well-structured participative decision making process among the different stakeholders.

2.2.2 Main Contribution

The main contribution of this research work is:

The design of a systematic methodology, called MULTIPAR, for performing HW/SW partitioning of embedded applications on heterogeneous platforms. This methodology is able to find solutions that are the result of decisions taking into account many system requirements, project management constraints and business goals.

Specifically, as an outcome of this research work, we provide:

- A formalized partitioning process flow. The main features are the following: (i) enabling the application deployment into HW and SW components based on multiple criteria decisions, which take into account multiple EFPs related to runtime and lifecycle, project management/business goal-related requirements and constraints; (ii) enabling decisions based on properties of existing and newly-designed component variants; and (iii) supporting the stakeholders in explicitly taking into
account ethical aspects while performing partitioning decisions. This contribution is discussed in Papers I, IV, V and VI.

- **The definition of a metamodel for describing HW and SW components.** We provided a formal model of a component-based system (CBS) that allows the specification of both HW and SW components and their EFPs. The metamodel is used in MULTIPAR as support for handling the repository of existing components and to enable the partitioning using MCDA methods. This contribution is presented in Papers I and II.

- **The categorization of HW and SW component EFPs with respect to the partitioning.** We conducted an analysis of EFPs with respect to HW and SW component deployment, which resulted in a categorization of the EFPs with respect to the HW/SW partitioning. Based on this categorization, through interviews and surveys we assessed the impact of the EFPs with respect to the partitioning decisions in the automation and control domain. This contribution is presented in Paper II.

- **A set of composition rules to analytically derive system EFPs from component EFPs.** We categorized the system EFPs and proposed a set of rules to analytically derive system EFPs from the component EFPs. This contribution is presented in Papers IV and V.

- **A suitability assessment of MCDA methods and related tools for the HW/SW partitioning.** Our main objective was to investigate the suitability of MCDA methods and tools for partitioning. This contribution is presented in Paper III.

In addition we have developed a tool that implements the proposed methodology. This contribution is part of Paper V.

### 2.2.3 Contribution through the Included Papers

A more detailed description of the contributions is provided here through the presentation of the peer-reviewed journal and conference papers, and the technical report included in this thesis.

**Paper I**

Chapter 2. Research Overview


**Short Summary** - This paper introduces MULTIPAR, a new methodology for performing HW/SW partitioning decisions of embedded applications. MULTIPAR combines model-/component-based techniques and MCDA. The major results of this paper are a formal definition of an architectural metamodel for specifying component-based applications and a decision process flow for performing the partitioning. The metamodel proposed here, extends the specifications of existing similar metamodels (e.g. [12, 13]) in order to describe both SW and HW components. It allows the specification of EFPs of component variants and it is based on the transition of system requirements and development constraints into systems and component properties. We demonstrated the approach in an industrial context.

**Own Contribution** - I was responsible for the detailed specification of the metamodel and the definition of the process flow activities for enabling the partitioning. In addition, I showed the applicability of the proposed metamodel on an industrial application.

**Paper II**


**Short Summary** - In this paper, the specifications of the metamodel presented in Paper I are further extended. Specifically, in order to manage different types of EFPs for both HW and SW component variants, the component property metamodel was further extended and specified in detail. From the literature and state of practices analysis a categorization of the EFPs in relation to HW/SW partitioning was provided. The EFPs are clustered into three main categories: lifecycle, runtime and project/business-related. Using this categorization, we performed a survey and interviews with 15 experts. The main aim of the survey was to investigate the impact that HW/SW partitioning has on the EFPs.

**Own contribution** - I was responsible for the modeling of the component-based system and of the component EFPs, the categorization of partitioning-
related properties, the design and realization of the interviews and surveys as well as the analysis of the results.

**Paper III**


**Short Summary** - The focus of this paper was to analyze the suitability of MCDA techniques for HW/SW partitioning. The main goal was to identify the most suitable existing methods and tools to support MULTIPAR. The paper describes a survey of the most well-known MCDA methods and tools for a specific class of MCDA methods. This class is called Multi Attribute Decision Making and deals with a finite number of possible alternatives solutions. Based on the literature and state of practices studies, the paper defines 11-suitability criteria which aided us in analyzing the appropriateness of MCDA methods and tools for partitioning. The analysis was performed among 100 MCDA methods and tools, and showed that a method or tool does not exist that is able to satisfy all of the “11-suitability criteria”. Hence there is no method or tool that can be directly used for the partitioning. However, the results showed the potential of using MCDA for partitioning decision processes and provided a starting point for further improving MULTIPAR.

**Own contribution** - I conducted the investigation of the MCDA methods and the related tools. Based on the literature and state of practice study, I defined the partitioning requirements and the “11-suitability criteria” which were of key importance in assessing the suitability of MCDA techniques for the partitioning.

**Paper IV**


**Short Summary** - This paper addresses the problem of the composability of EFPs at system level. Although in general this is not a solvable problem,
under typical constraints of embedded systems for many EFPs it is possible to analytically derive the system EFPs from the involved component EFPs. The paper proposes a set of composition rules related to runtime and lifecycle requirements and project management constraints that can be applied to calculate EFPs starting from component EFPs. The applicability of these composition rules is demonstrated through an industrial case.

**Own contribution** - I led the overall work and my main contribution was the definition of the composition rules that can be used to calculate the system EFPs, and demonstrated their applicability via an industrial case. I also contributed to the classification and the definition of the assumptions under which the composition rules are valid.

**Paper V**


**Short Summary** - This paper proposes an extension of MULTIPAR to optimize system EFPs. The MULTIPAR process is here augmented with activities and artifacts that allow systems EFPs to take into consideration when carrying out the partitioning. This paper also presents a tool which implements the entire MULTIPAR methodology. The main purpose of this tool is to support designers when performing the partitioning by applying MULTIPAR. This paper describes the validation of the entire MULTIPAR process and the system composition rules for the runtime EFPs proposed in Paper IV. The validation was performed through an industrial case.

**Own contribution** - I proposed the extension of the MULTIPAR process flow in order to take into account system EFPs. I also designed the high-level architecture of the tool implementing MULTIPAR and guided its realization. I managed the validation of the system EFP composability formulas for the runtime properties, and I validated the feasibility of the entire MULTIPAR approach. For performing the validation, I developed an industrial application which was used to demonstrate the feasibility of the methodology and the applicability of the proposed tool.
2.3 Research Approach and Validation

In this section we describe the overall research approach and how our research work was validated.

2.3.1 Research Methods

To perform our research, we followed the guidelines proposed by Shaw in [14, 15]. Our research strategy is depicted by Figure 2.1 and described as follows:

- Definition of the Practical Problem: the main research problem is the lack of an approach that systematically helps engineers bring HW/SW partitioning decisions based on many and different types of EFPs.

- Definition of the Idealized Problem: for our idealized problem we defined a few assumptions - the embedded systems are component-based systems in which components are implemented either as HW or SW execution units. We focused on the automation and control domain where
applications typically use simple operating systems and have several constraints that enable simpler composition of EFPs.

- Development of Research Products: the main outcome of our research is the construction of a new partitioning methodology whose main features were described in Section 2.2. It provides a set of specifications (e.g. metamodel, process flow, guidelines, formulas) for solving the idealized problem. The proposed specifications are implemented in a tool which implements the process flow, manages the required artifacts and provides the partitioning results.

- Research Validation of the Idealized Solution: we validated the proposed idealized solution using several empirical methods. Specifically we used industrial cases, interviews and surveys. All methods were applied in an industrial set-up. Additionally, we designed a testbench to perform physical measurements of the system and component EFPs of the industrial cases.

The Solution and Validation of the Practical Problem is not part of this doctoral thesis’ scope.
2.3 Research Approach and Validation

2.3.2 Validation

The validation was performed progressively throughout the entire doctoral research period.

The main question leading our validation was:

*Is the proposed methodology feasible in an industrial context?*

The main objective of our validation strategy was to demonstrate that MULTIPAR provides partitioning solutions that are applicable to industrial systems.

To reach our objective we conducted surveys and used two industrial cases. The first case was related to the development of a small wind turbine controller within the context of the iFEST project [16]. Through this industrial case we demonstrated the applicability of MCDA for performing the partitioning. In this study, the selected partitioned solution was the result of considering only EFPs of the components while system EFPs were not taken into account. The second industrial case was focused on the realization of a direct current motor control application which includes safety functions. It was developed within the context of the CLERECO project [17]. Through this second case, we validated the complete methodology, i.e. the best traded-off partitioning solution was the result of taking into account both component and system EFPs.

It is worth to mentioning that part of our strategy was centered around the following two key concepts of MULTIPAR i.e. the MCDA-based partitioning and the EFP handling:

- **MCDA-based partitioning**. Here, our focus was first to answer the following question: *Is MCDA suitable for HW/SW partitioning?* Through the review of the available literature and surveys carried out with senior practitioners we collected the data needed for the suitability assessment of the MCDA methods and tools. We performed a survey involving 108 MDCA methods and 22 tools to assess MCDA suitability for partitioning. The suitability analysis showed that only a few methods had the highest potential for being applied in partitioning processes. In particular, the most suitable candidate was the Evidential Reasoning (ER) method [18] which allows to address complex decision problems dealing with quantitative and qualitative criteria under uncertainties and randomness. The results of the suitability analysis were used in both industrial case to confirm the applicability of these methods for partitioning. Then, our focus was on answering the following question: *Can the partitioning
solution achieved via the MCDA methods be executed on the given platform? We answered this question through the industrial cases, i.e. we deployed and ran the partitioned solutions into the given heterogeneous platforms.

- **EFP handling.** Here, our objective was to establish which EFPs depend on partitioning and to what extent these EFPs are dependent on HW/SW deployment. To achieve this we designed and performed a survey and many interviews. In addition, we aimed to validate the proposed set of formulas to calculate the system EFPs. In particular we were interested in analyzing the system EFPs related to the physical behavior of the application (i.e. execution time, power consumption, memory footprint and FPGA area). To achieve this, we posed the following question: *To what extent can system EFPs be properly composed from the component EFPs?* This led us to perform a literature review, and an analysis to evaluate which of the existing and available methods and tools were more suitable for measuring the EFPs under analysis. Subsequently, we used the second industrial case to measure the EFPs at component and system level. The results showed that solutions achieved by applying MULTIPAR in combination with the composition the proposed formulas are sufficiently close to the solutions where the measured values of EFPs have been used. All of the experimental measurements are repeatable.
Chapter 3

Related Work

The related work is divided into three main research topics: (i) HW/SW Partitioning, (ii) use of MCDA, and (iii) EFP Modeling and Composability for Embedded Systems.

Hardware/Software Partitioning.

In literature, HW/SW partitioning is a widely discussed research topic in the context of the HW/SW co-design discipline, e.g. [19–23]. HW/SW partitioning is considered to be one of the main challenges faced when designing embedded applications: “as it has a crucial impact both on the cost and on the overall performance of the resulted product” [24]. In literature the partitioning problem is stated as “finding those part’s of the model best implemented in hardware and those best implemented in software” [19] or “choosing the software and hardware implementation for each component of the system specification” [23]. In [25] the HW/SW partitioning is defined as “the problem of dividing an application’s computations into a part that executes as sequential instructions on a microprocessor (the “software”) and a part that runs as parallel circuits on some IC fabric like an ASIC or FPGA (the “hardware”) to achieve design goals such as required performance, power, size, and cost”. HW/SW partitioning is considered to be a NP (non-deterministic polynomial)-hard problem [26].

There is a considerable body of knowledge related to the partitioning problem. It was intensively studied during the 1990s and the early years of the 2000s, when HW/SW co-design basically emerged as a new discipline [20]. Over the decades, a wide range of approaches have been proposed in order to
automate/support HW/SW partitioning using different strategies, for instance dynamic programming [27], heuristic algorithms based on the tabu search, simulated annealing, genetic algorithm techniques [28–31], integer programming [22], and multiple objective optimization techniques [32], [33]. An in-depth study of several partitioning approaches and related issues is provided in [31] and a walk through of the highlights of partitioning approaches over the last two decades can be found in [20]. These approaches are mainly oriented towards solutions satisfying performance requirements. They provide partitioning solutions whose results are derived from decisions which consider platform related indicators, like potential speedups, FPGA area, communication overheads, locality and regularity of computations [34]. Two examples are Vulcan [35] or COSYMA [36] which target optimization problems focusing on design constraints such as timing, or resource speed-up [37]. Even though a few partitioning approaches exist which optimize a combination of EFPs (e.g. design costs, energy consumptions, performance, etc.) [38,39], they are limited with respect to the total number of EFPs which they are dealing with. In addition, current partitioning approaches focus on technical characteristics such as resource utilization, and energy consumption and performance, and do not consider properties related to the project development and business perspectives (e.g. legacy, resource availability, design effort, testing cost, production cost, etc.). Neither do they consider EFPs important for dependable systems such as safety, reliability, and security.

Multiple Criteria Decision Analysis.

MCDA (also known as Multiple Criteria Decision Making (MCDM), or multiple criteria decision aid) is a sub-discipline of operational research, whose objective is to “support the subjective evaluation of a finite number of decision alternatives under a finite number of performance criteria, by a single decision maker or by a group” [40]. In [41], MCDA is defined as “an umbrella term to describe a collection of formal approaches which seek to take explicit account of multiple criteria in helping individuals or groups explore decisions that matter”. One of the most well-known surveys on MCDA is provided in [10]. MCDA is widely applied in several fields such as medicine, human resources management, pattern recognition, marketing, financial management and economics, environmental and energy management, and computer science [10,42] to solve complex decision problems. A general technical framework in addressing MCDA problems is described in [43]. Several MCDA methods exist and are presented in [40]. MCDA methods “have been designed in order to designate a preferred alternative, to classify the alternatives into a
small number of categories, and/or to rank the alternatives in a subjective order of preference’. The key aspect of MCDA is that it allows one to “take the multidimensionality of decision problems into account by using multiple criteria, instead of one common denominator” [44].

MCDA problems are roughly divided into two main classes [10, 45–47]. The first class includes methods dealing with problems which consider a finite number of possible alternative solutions (it is also referred to as multiple attribute decision making (MADM)) [48–50]. For this class of methods the alternatives are explicitly given via a well-structured decision-making process, which takes into account the stakeholders’ preference, allowing the ranking, and classifying of alternatives or the choice of a sub-set of alternatives [51].

The second class, known as multiple objective decision making (MODM) [45, 49, 52], considers problems dealing with an infinite number of alternative solutions. It addresses “decision problems in which the decision space is continuous” [47]. In this class of problem a set of objective functions are optimized with respect to a set of constraints [49].

The main advantages of MCDA solutions are presented and discussed in [49] [53]. A comparison highlighting the key difference between MODM and MADM classes can be found elsewhere [45, 49, 54], but we present two short lists underlining: a) the key concepts around both MCDA classes and b) the most relevant differences between these two classes. Since no global definitions of key concepts are available [55], authors use them interchangeably.

MCDA key concepts:

a) Attribute: defines the characteristics, qualities or performance parameters of the alternatives [56].

b) Alternative: represents the item to evaluate according to its attributes in order to find the final decision solution (or set of solutions).

c) Criterion: expresses a measurement of effectiveness representing the basis for evaluation [57].

d) Objective: is defined as “reflections of the desires of the decision maker” and it “indicates the direction in which the decision maker wants the organization to work” [56].

e) Goal: (or target) can be interpreted as “apriori values or levels of aspiration” that can be “either achieved or surpassed or not exceeded” [57].
Chapter 3. Related Work

The main differences between MODM and MADM:

a) In MODM, the criteria are defined by objectives, while in MADM, the criteria are defined by attributes.

b) When applying MODM methods the objectives have to be explicitly defined, while in MADM they are implicitly defined.

c) In MADM the alternatives are explicit and finite, while in MODM, the alternatives are infinite and implicitly defined.

Our main interest was to find a method able to deal with a large number of EFPs, which can be heterogeneous. As a consequence we directed our studies towards finding a MCDA method (or collection of methods) suitable for enabling partitioning based on several criteria of different nature. The criteria in our approach are derived from the EFPs related to the application requirements and project/business-related constraints. Over the last few decades, a large number of MCDA methods has been proposed, but they differ in the realization of multiple criteria [54], e.g. objective, criteria-assessing, weight computation and preference model, and algorithms applied. Hence, depending on the decision-making user context and the type of problem to be solved, some methods are more suitable than others for reaching certain goals. Several solutions have been proposed to assess the appropriateness of MCDA methods for solving decision-making problems, e.g. [54, 58] in which different methods are assessed with regard to sustainability issues, or [59], where the methods are assessed for their suitability in solving seismic structural retrofitting issues. In [54, 58] a similar approach to our work in investigating the suitability of MCDA is taken; however, they deal with different application domains. Hence, only a few “core” concepts (such as criteria interdependence, process transparency, etc.) can be reused in our research.

When considering the MCDA in relation to the HW/SW partitioning state-of-the-art, previous work [20, 60–62] applied MODM methods for high-level codesign exploration. Our approach, however, is based on MADM methods. In addition, our approach brings together project-related constraints and business goals (together with lifecycle as well as runtime non-functional requirements) [63] into the set of decision criteria. For instance, in [64] the problem of design space exploration at system level is tackled by using a multi-objective pseudo boolean solver. However, the partitioning is applied on a limited number of EFPs (specifically, for the test cases, the analysis is limited to just two EFPs related to the run time behavior: the area and the power consumption). The
problem of selecting the solution is left to the decision makers/designers, who have to select it from a continuous spectrum of possible solutions in the Pareto-front. In our approach, we take into account a large number of EFPs, and the partitioning solution is selected from a finite number of possible HW/SW configurations.

While the authors of [65] introduced the concept of scenario-based design space explorations, the mapping is performed by either using a multi-objective genetic algorithm method, or by using dynamic sequential oscillation search methods, or by applying a combination of both. However, this leads again to the problem of selection from a continuous space of possible solutions, hence this approach is only applied to a limited number of objectives (i.e. performance, energy and cost).

In order to decompose the multi-objective optimization problem into subproblems, a hierarchical top-down optimization methodology is proposed in [66]. Nonetheless, this suffers from the exact selection issues mentioned above.

**EFP Modeling and Composability for Embedded Systems.**

The management of EFPs in component models as well as the composition of EFPs are key challenges in the component-based software engineering community [13]. There are several component models that are specified by means of metamodels, for example Palladio [67], Pecos [68], Save_CCM [69], ProCom [13, 70] which in a similar way define interfaces and EFPs.

Some component models allow different implementations of components, but they assume SW components, which have the same EFP types for all variants, though with different values. The “management of EFPs in component models is one of the main challenges in the component-based software engineering community” [13]. The management of extra-functional requirements in embedded systems development has been tackled for instance in [71], [72], and [73]). In [74–76] techniques to analyze properties typical of embedded systems have been presented (e.g. related to the analysis of timing properties). In [77–79] approaches to estimate the EFPs (such as, for instance, energy) for SW and HW implementations have also been proposed. The runtime behavior of a HW implementation on a FPGA, for example in [80], is estimated by exploiting the simulation and a performance model of the FPGA. A statistical approach is proposed in [81] in order to estimate the execution time of embedded software. However, in the recent decades HW and SW design methodologies have become more alike [19], and, as already foreseen in [5], they require designers to have a unified view of SW and HW, which converges with the concurrent design of HW/SW [19] (see, for example, the HWSWCO
We have extended the current research state-of-the-art by providing a meta-model capable of (i) describing embedded component-based applications for both HW and SW component implementations, and (ii) capturing systems EFPs (from lifecycle and runtime perspectives) and project/business related EFPs. Specifically, our approach allows the description of components independently of the underlying platform and allows the management of EFPs with respect to different component variants.

Another direction of interest for our study is the system EFP composability of the HW and SW components involved. Despite the fact that this topic has already been addressed in multiple studies in the past (as in for instance [84, 85]), composing system EFPs is still a key challenge when developing software component-based systems [86,87]. For timing analysis a compositional timing analysis for resource-sharing systems is proposed in [88], while [89] proposes a response time analysis for fixed priority preemptive systems. In contrast to our approach, these works do not use the existing EFPs values of the components considered in isolation but they model the entire system for solving the specific problem. Though the EFP composability problem was intensively studied from the functional and non-functional perspective (for example in [86, 90, 91]), to the best of our knowledge, no previous work has been carried out taking a SW and HW combined perspective. For HW/SW co-design, there are studies that address component compositions (e.g. [92]) via their modeling platform (such as Ptolemy II [93]), but the composition is based on the analysis of functional and timing properties only.

In contrast to these works, we took into account the strong restrictions and assumptions on the execution context typical of embedded systems, which allows us a to formalize a set of composability rules for many EFPs.
Chapter 4

Conclusion

This chapter concludes the thesis through a short summary and a brief description of how the research questions have been answered. It also introduces future directions towards a possible continuation of this research work.

4.1 Summary

The main objective of this research work is to provide a solution for HW/SW partitioning of embedded applications that should allow partitioning decisions to be performed based on many EFPs. We target this objective by the design of MULTI{PAR}, a new HW/SW partitioning methodology for component-based applications. MULTI{PAR} provides a metamodel which allows the specification of both HW and SW components and their EFPs, and a set of rules to compose system EFPs from component EFPs. It applies MCDM methods to make the decisions based on many EFPs, which are related to the runtime, lifecycle and development process. It also provides an assessment of MCDM methods that are suitable for being applied for partitioning. Moreover, it defines a process which starts with the identification of the EFPs that are of interest for the partitioning. Based on the identified EFPs at component and system level, two optimizations are performed by applying MCDM. The first optimization leads to the selection of the most suitable HW and SW variants for each component by using MCDM in order to reduce the space of the design. Subsequently, based on the results of the first optimization, all system variants (i.e. combinations of the selected HW and SW variants for each component) are ranked
by applying MCDM. The partitioning decisions for carrying out the selections will be made in a late stage of the design phase. This process will supposedly lead to an optimal partitioning configuration (i.e. system variant), which meets the application development goals. As part of our research work, we also suggest a way to explicitly consider ethical aspects during the partitioning decision process. We validate the feasibility of MULTIPAR on industrial demonstrators.

4.2 Remaining challenges

MULTIPAR has some challenging points. The first challenge is related to the efforts devoted to the entire process. Quite some effort is required to determine the specifications and measurements of EFPs of interest for the partitioning. The effort is strictly related to and proportional to the architectural decisions, so which EFPs and what level of accuracy is needed are a consequence of the initial architectural decisions. The effort will increase with the increase of the complexity of the architecture. For simple applications and few EFPs of interest, the process is simpler and more straightforward.

The second point is connected to the accuracy of the EFP values. For many component EFPs, in particular for newly designed components, the degree of accuracy could be different. It may come about that the accuracy of a particular component EFP value is different when it is related to a component that has already been implemented compared to when it is a new component where the EFP value will be estimated. For this reason it could be useful to implement a sensitivity analysis with variations of EFP values.

In MULTIPAR it is not precisely defined when a newly designed component should be developed. It is assumed that a new component will be implemented immediately after the system variant selection. However, the process will not require a measurement of the component EFPs later implemented. For future reuse of the newly implemented component a measurement of the EFP values might be included as an activity to improve the MULTIPAR process.

Another challenge is related to the set of composition rules provided, which are expressed as linear combinations, and there is no consideration for dependency between the EFPs, or dependency between component EFPs in respect of their deployment. These challenges are partially solved by relying on the constraints valid for the application domain taken into account (i.e. the control and automation). The industrial cases have shown that the assumptions hold, however, in general, these assumptions might be violated.

Last but not least is the challenge related to the fact that MULTIPAR is
a top-down approach which starts from the requirements analysis. It is not elaborated how MULTI PAR can work in an agile (or incremental) process in which the requirements can be changed or new requirements can appear during the development process. However, when the way to calculate or estimate the EFPs of interest is defined, the MULTI PAR process can quite easily be reiterated. This might be the case when a requirement or constraint needs to be either added or changed.

4.3 Answers to the Research Questions

Here, we revisit the research questions and briefly discuss the answers to them.

RQ1. What would be a systematic decision process to support engineers in performing HW/SW partitioning of embedded applications?

To answer this research question we applied MCDM, CBD and MBD methods and combined them into a new methodology, MULTI PAR. MULTI PAR defines a partitioning decision process, which is specified through a set of activities and artifacts. Starting from the application architecture, and the application requirements, the architect(s) start to identify the component and system EFPs of interest. From the repository of existing component variants, they select the possible candidates. Where needed, the values of the component EFPs have to be calculated or estimated for both existing and newly-designed components. Subsequently, engineers apply MCDM to find the best HW and SW component variant candidates with respect to the given goals. For each component the best HW and SW variants will be used to build all possible system configurations (i.e. system variants). For each system variant the system EFP values have to be provided. Some of them can be calculated using the composition rules that we proposed. Lastly, MCDM will be carried out one more time in order to find the most suitable system variant to deploy on the platform.

RQ2. Which EFPs depend on the HW/SW partitioning, and in which way?

We categorized the EFPs into three main classes: runtime, lifecycle and project/business goal-related. Our work has shown that runtime EFPs clearly depend on the HW/SW partitioning, but also lifecycle EFPs have a clear impact on partitioning. As a general consideration, HW solutions provide better runtime EFPs related to system performance, while SW solutions provide bet-
Chapter 4. Conclusion

The characteristics of lifecycle EFPs. Our research work has also shown that project/business goal-related EFPs are important for partitioning decisions. For this type of EFP, SW implementations are preferred.

RQ3. What methods and related tools are suitable for specifying and comparing the EFPs in the HW/SW partitioning process?

We focused our research on MCDA, and specifically on MCDM. We performed a suitability analysis as a result of which we identified that the ER method is the most suitable for being applied to HW/SW partitioning. However, there are other methods such as, for instance, NAIADE [10], PROMETHEE I and II [10], REGIME [10] and EVAMIX [10], that can be used for partitioning as well. ER is also implemented in a tool called Intelligent Decision System (IDS) [94], while PROMETHEE methods are implemented in a tool called Visual Promethee [95]. Both tools have been assessed as the most suitable ones for partitioning.

4.4 Future Work

Here we present some considerations for future research directions. The first consideration is related to the way in which human preferences and judgements are handled in MULTI PAR. In MULTI PAR we suggested a few techniques for assigning the decision criteria weights (e.g. SMART and its improvements SMARTS, SMARTER [96, 97]), however, we envisage the need to complement MULTI PAR with additional activities which are able to better mirror the weights of all of the stakeholders’ concerns and judgements. We also see the need for handling them in a more explicit way. Furthermore, in order to provide additional support to engineers when performing the partitioning, we see the necessity to encapsulate in MULTI PAR a method for performing sensitivity analysis. This will allow the assessment of the impact of uncertainties in the EFP values for a given partitioning solution.

Both of the aforementioned improvements could be implemented in the MULTI PAR tool.

In relation to the handling of the EFPs, we see two possible research directions to be addressed. The first one is related to the specialization of the EFP categorization. We see the possibility of including, in the categorization, other standards such as, for instance, the ISO/IEC/IEEE 42010 international standard [98]. In addition, it might be of interest to conduct an impact analysis...
of the EFPs in relation to the partitioning in other domains (e.g., automotive, robotics, etc.) in order to complement the guidelines provided for the automation and control domain. The second research direction is related to the handling of EFPs that are non-composable. Examples of these properties are safety and security. Today, the engineers who are carrying out the partitioning using MultiPar have to specify/calculate these types of system EFPs manually for each possible system variant configuration. This approach is quite infeasible when a large number of system configurations have to be considered.

An interesting future validation of the proposed methodology would be to investigate the applicability of MultiPar in several development processes with the aim of assessing firstly if MultiPar is more efficient than current partitioning approaches, and consequently if the efficiency of MultiPar increases when more existing component variants are available in the repository.
Bibliography


II

Included Papers
Chapter 5

Paper I: Modelling for Hardware and Software Partitioning based on Multiple Properties

G. Sapienza, I. Crnkovic, and T. Seceleanu.

Abstract

In many embedded systems types the separation process for deploying the applications as software and hardware executable units, called partitioning is crucial. This is due to the fact that partitioning decisions impact the overall life cycle of the systems. In industry it is common practice to take partitioning decisions in an early stage of the design, based on hardware and software designers expertise. We propose a new methodology as a combination of model-based and component-based approaches which enables a late partitioning decisions based on high level system requirements and project constrains. The final partitioning is decided based on a multi-property analysis approach. Here, we focus on the formalization of the overall process and in particular on the definition of a comprehensive system metamodel. This is meant to support modelling approaches suitable for enabling both the partitioning and reuse. An industrial case study is used to illustrate the approach.
5.1 Introduction

Many modern embedded systems are implemented as software (SW) and hardware (HW) components, i.e. deployed as SW on CPU platforms and as HW programmed in Field-Programmable Gate Array (FPGA). The heterogeneous platform dramatically increases the system performance. However, the increasing complexity of industrial embedded applications constantly challenge designers when making decisions upon the separation into HW and SW for their deployment. These design decisions are of crucial importance since they impact (i) the application performance and its quality, (ii) the entire development process, and (iii) system lifecycle management. They require to be taken upon several criteria which are derived from a number of requirements and project constraints, which differ, vary and have even conflicting priorities. Like in other architectural decisions, the partitioning (i.e. the SW/HW separation process) requires a trade-off analysis. Over the decades, several techniques and approaches have been proposed for SW/HW partitioning, but they were mostly driven by the technological advances in electronics performance [1], [2], [3] and often tackled the optimisation problem with focus on performance and costs [4], [2]. In industry, mainly due to time pressure imposed by the time-to-market and cost optimization, it is a common practice to take partitioning decisions in an early stage of the design phase and without the support of systematic approach. They are most likely based just upon HW and SW team expertise, which often is not synergetically combined [1]. The decisions are lacking a support obtained by a comprehensive exploration of different options with respect to requirements and project constraints. Overdesigns, underdesigns, redesigns, flow interruptions in design and implementation, unplanned iterations are some of the typical drawbacks generated by this practice [5].

This paper addresses these needs by introducing a new approach Multiple Criteria-based Partitioning Methodology, we called MULTIPAR.

This approach aims for (i) providing partitioning decision in a late stage of system design, and (ii) base the partitioning decision on many and diverse criteria. The main contributions in this paper are a) formal definition of a comprehensive metamodel able of describing both SW and HW units and b) description of the MULTIPAR process flow which is based on the following strategy: (i) enabling application technology-independent design in the early stage of the design phase and pushing the partitioning decisions for enabling platform-specific design to a late stage, (ii) enabling the application deployment into SW and HW based on multiple criteria decisions which account for the high level systems requirements and project constraints, and (iii) making also de-
cisions based on a reusable set of alternatives, in order to enable design and implementation reusability.

We illustrate a model conforms to the metamodel and the basic concepts of MULTI-PAR on an industrial case: a simple wind turbine control system. The rest of the paper is organized as follows: Section 2 presents the metamodel. Section 3 defines the MULTI-PAR process, Section 4 illustrates a case study, Section 5 presents the related work and Section 6 concludes the paper and provides the future directions of this research work.

5.2 The Metamodel for Partitioning

Component models and metamodels. There exists several component models that are specified by means of metamodels, for example Pecos [6], and ProCom [7] which in a similar way define interfaces and Extra-Functional Properties (EFPs). Our approach extends these specifications by allowing variable management of EFPs with respect to different component variants. By allowing a variable set of EFPs for component variants we have made it possible to reason about very different implementations, like software and hardware, and we have made it possible to easier compare the exhibited properties with those that are required.

In order to formally and unambiguously capture the essential domain concepts for components that have to be partitioned into SW or HW and the relevant relationships among them we have developed a metamodel. Figure 5.1 shows a simplified overview of the diagram describing the proposed Component-based System (CBS) metamodel. It is also used along the following subsections to explain the key concepts of the metamodel. Each CBS is supposed to be identified, described and documented as shown in Figure 5.1 by the Entity Identification entity.

Component-Based System. The central core is represented by the Component BasedSystem entity which compliantly to the definition provided in [8]. It is composed by a platform, a set of both components and connectors.

Platform. The Platform entity is meant to abstract the model of an underlaying and already defined platform, on which an application is deployed in form of SW and HW components. It consists of PlatformComponent and Platform-Connector entities.

A PlatformComponent entity models either a digital or analog HW component or a low-level SW component. The PlatformComponentType entity is used to indicate if the PlatformComponent is an HW or SW. The application
components are deployed on platform components. This is defined by the semantic relationship between the PlatformComponent and the Component entities named “to be deployed on”. Examples of hardware platform components are: CPUs (single and multi-core), FPGAs, RC-analog filters, etc. Examples of SW platform components may be considered: RTOSs, Drivers, Ethernet Stacks, etc.

A PlatformConnector entity serves to model the connections between platform components. It can be HW or SW, as modelled by the PlatformConnectorType entity. Examples of platform connectors are: CAN bus or Ethernet. In order to be bound by a connector, platform components have to be able of playing two different roles: named Target and Source. If a platform component plays both roles, it can be bound by the same connector. This is, for instance, an operational amplifier or flip-flop when connected in feedback.

Component. A Component entity is meant to abstract the model of an application component. In order to support MULTIPAR, a component consists of an Interface and one or more Variant entities.

The Interface entity models the functional properties of the component, i.e. the component interface is divided into the required and provided interfaces. The component interface is defined through a number of ports. In Figure 5.1, a port is represented by the Port entity. It is modelled by assuming that it can play two roles, named InPort and OutPort, which serves to respectively model the
required and the provided interfaces. It is also assumed that each component at least has an input port. A port is also characterized by the mode in which information is exchanged, for instance synchronously or asynchronously.

The Variant entity models a deployed component on a platform. Each component may have more variants associated with it. Each variant is characterized by an implementation technology (i.e. SW or HW), a format (i.e. C-code, VHDL-code, binary, etc.), and a platform of deployment (i.e. the operating system, the processor technology, etc.). A variant is specifically defined by the relationship with the data type associated to each port. Although the interface is the same, regardless of the component variants, each variant is characterized by a specific platform-dependent implementation, resulting into a specific port implementation. This is shown in Figure 5.1 by the PortTypeBinding and PortDataType relationships. Examples of PortDataType are: Single Analog (e.g. ADC) Single Digital (e.g. I2C, RS232, etc.), Multiple Digital (e.g. 8/16/32-bit BUS, etc.), etc. Further, each variant has associated with it a number of EFP entities, meant to model the component’s EFPs. An EFP entity is characterized by: (i) a category e.g. component execution time, component size, component reliability, etc.; (ii) a subcategory, which might be needed to distinguish EFPs specific for HW or SW components (e.g. for the component size category, examples are: the memory footprint for a SW component and the number of gates for a HW component); (iii) the context under which the property value is set; (iv) a collection of values for the given property.

Connector. A Connector models the binding between the components. The bindings are realized through the interfaces. Thus, the connector always binds two ports. These must be an input and an output port respectively, even if belonging to the same component (e.g. for implementing feedback signals). If the same component is connected through an input port and output port, it must be able to play both roles, i.e. the Source and the Target - Figure 5.1. An output port can be associated to more than one connectors.

5.3 The MultiPar Process

The MULTIPar process includes a set of activities which lead to the HW/SW partitioning of the application components. The process adopts a Component-based Development (CBD) process with emphasis on components’ reuse. The MULTIPar process is performed in several iterations of two phases: the first one building the system architecture with focus on the functional requirements and the second one considering extra-functional requirements.
During the process several artifacts are used as input and some new artifacts are being created. A list of application requirements and a list of project constraints are used as input to the analysis, design and architecting activities. In addition, information about existing components is used, from a component library. The components in the library are compliant to the metamodel defined in Figure 5.1.

The MULTI PAR process flow described by Figure 5.2, embodies the following activities:

Figure 5.2: MultiPar Process Flow Diagram.
Application Modelling and Component Selection. The modelling activity is based on the information provided from the application requirements and project constraints. By using this information as inputs, the application is modelled as a set of platform-independent components and connectors binding the components, compliantly with the metamodel defined in Figure 5.1. If found in the component library, the defined components will be inserted into a matrix called Decision Matrix (DM). In this process all available implemented component variants \( C_i \) will be included. They will have different implementations and different properties, but the same interface. The architectural design and components’ identification is a complex process and may require several iterations. For each identified component which do not exist, a new component entry will be inserted into the DM. This component entry will include the interface specification and two virtual (i.e. yet non-existing) component variants (a SW one and HW one).

Component Required Properties Identification and Prioritization. After the architecture design and component identification, the component properties that fulfill the extrafunctional requirements must be identified. From the extra-functional requirements and project constraints and by the architectural analysis two sets of properties of interest for the partitioning decisions - application properties \( P_{iA} \) and project-related properties \( P_{iP} \) will be identified and their related values will be defined. These are the required properties of the identified components. The required properties should be prioritized based on a trade-off analysis. The priorities will be used as weight factors for the MCDA. The outcomes of this activity will be saved in the DM. The component variants \( C_i \) have some properties, i.e. they exhibit some properties. We define them as exhibited properties \( P_{iE} \). The sets \( P_{iA} \) and \( P_{iP} \) may include some properties that are not defined in the component variants \( C_i \). The missing values will be setup in a late stage.

Component Variants Filtering. Many of the component variants do not satisfy the specifications of the required properties. For example, there can be a required property that a particular component should be deployed on a particular component platform or an EFP, for example memory usage, is far above the memory allowed to be used, as specified by the corresponding required property. Such variants are not relevant for the application and can be discarded as a candidate, i.e. they can be removed from the DM. The filtering activity is performed whenever the values of the exhibited properties are changed.

Defining Exhibited Properties Values. For the components variants which the \( P_{iE} \) do not have the specified values, the values have to be assigned. They might be measured, simulated or estimated. They will populate the DM (and
5.4 Wind Turbine: Industrial Case Study

possibly the component library for future reuse).

**Multiple Criteria-based Component Partitioning.** When the exhibited properties and the required properties are defined as well as the cost function expressed by the required properties priorities it is possible to search for a (local) optimal partitioning solution. Different MCDA techniques can be used in the solution provision. This activity might converge to a single or several solutions or no solution can be found. In a case it converges to a single solution the process flow is concluded. If several solutions are available, they will be ranked. If it will not converge new iterations need to be carried out.

**Solutions Ranking.** This activity is the last one and is supposed to be performed in case several partitioning solutions are available. Based on project constraints and application requirements, further decision criteria will be defined for enabling MCDA-based ranking.

Figure 5.2 does not include the analysis and verification activities; they are iteratively performed along to each of the above described activities.

5.4 Wind Turbine: Industrial Case Study

![Image of Wind Turbine Application (WTA) Model]

Figure 5.3: Wind Turbine Application (WTA) Model.

To show the conformability of a model from the proposed metamodel and the feasibility of MULTI\textsuperscript{PAR} in an real industrial application we have developed a simple wind turbine application (WTA). The main objective of the WTA is to control the transformation of the rotational mechanical energy of the rotor blades, caused by the wind, into electrical energy, which will be redistributed via a power network. The core element of the application is the controller, which dynamically regulates the rotor blades at different wind profiles while maximizing the generation of electrical energy and avoiding any damage to the plant. The application is deployed on an industrial wind turbine prototype.
In this case study we illustrate two simple different deployment scenarios of the same WTA. Both scenarios are driven by different sets of required properties. They are named respectively: **Performance-driven scenario** and **Effort-driven scenario**. The first scenario is originated by the need of satisfying application requirements on components execution time. The second scenario needs to satisfy properties derived by project constraints such as the effort for the technology-dependent design of components, adaptation of existing components, the testing, etc.

The WTA is characterized by a set of platform project constraints, one of them specifying a combined technology implementation: CPU and FPGA. A solution to this comes in the form of the Xilinx ZynQ chip.

**The Wind Turbine MultiPar Process Flow.** We briefly describe in the following the process flow behind the realization of the use case.

1. **WTA Modelling and Component Selection.** Based on the WTA requirements and project constraints lists, as well as on the information available from the existing components, the application architecture was defined. It is modelled as a number of interconnected components. The model is conformed to the proposed metamodel, and it is implemented by using The MathWorks Simulink. The simplified model is shown by Figure 5.3. The application is decomposed as follows: the **SensorInterface** (C1), interfacing the feedback signal coming from the sensors to the controller, i.e. the turbine speed (TS) and wind speed (WS) signals; the **Filter**, filtering the feedback signals (C2), the **Main Controller** (C3) orchestrating the overall control of the application; the **Pitch Estimator** (C4) estimating the desired pitch angle at the rotor blades; **Pitch Regulator** (C5) regulating of the pitch angle based on the desired pitch and the calculated pitch; **Park and Brake Controller** (C6) setting the pitch command for steering the rotor blade; and **Supervision System** (C7) supervisioning the execution of the overall application. Each component defined by its interface: input and output port, and ports are bound via connectors. For instance, the C3 interface is defined by the **Filtered_TS inport** and the **Pitch_Reference outport**. Example of connector, is given by the **Pitch_Brake_Cons** connecting C3 and C6. Each component has also an entry in the DM, as shown in Figure 5.4.

In order to validate the design of the application, the model has been simulated, using the plant model (represented in Figure 5.3 by the grey boxes) which is calibrated against a real wind turbine prototype.

The component library includes a set of variants (i.e. reusable components) for the components C1, C2, C3 and C4 and they are directly populating the DM (see Figure 5.4), whereas for new components, in this case C5, C6 and C7 two virtual variants: HW_v and SW_v are associated and the related EFPs values
5.4 Wind Turbine: Industrial Case Study

<table>
<thead>
<tr>
<th>C_ID</th>
<th>Instance Type</th>
<th>Exec Time (sec)</th>
<th>Development Effort (man/week)</th>
<th>Performance-driven Max Exec Time</th>
<th>Performance-driven Selected Instances</th>
<th>Effort-driven Max Design Effort</th>
<th>Effort-driven Selected Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1.1</td>
<td>SW</td>
<td>12µs</td>
<td>2mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1.2</td>
<td>HW</td>
<td>7µs</td>
<td>2mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1.3</td>
<td>HW</td>
<td>9.2µs</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2.1</td>
<td>SW</td>
<td>25µs</td>
<td>2mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2.2</td>
<td>HW</td>
<td>12µs</td>
<td>2mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2.3</td>
<td>HW</td>
<td>14.4µs</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3.1</td>
<td>SW</td>
<td>11ms</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3.2</td>
<td>HW</td>
<td>16.3ms</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4.1</td>
<td>SW</td>
<td>23µs</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4.2</td>
<td>SW</td>
<td>34µs</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4.3</td>
<td>SW</td>
<td>17.9µs</td>
<td>3mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4.4</td>
<td>HW_v</td>
<td>12.5µs</td>
<td>5mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C5.1</td>
<td>HW_v</td>
<td>13µs</td>
<td>5mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C5.2</td>
<td>HW_v</td>
<td>15.5µs</td>
<td>4mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C6.1</td>
<td>HW_v</td>
<td>11µs</td>
<td>7mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C6.2</td>
<td>HW_v</td>
<td>16µs</td>
<td>4mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C7.1</td>
<td>HW_v</td>
<td>10ms</td>
<td>10mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C7.2</td>
<td>SW_v</td>
<td>8ms</td>
<td>5mw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.4: Wind Turbine Decision Matrix (DM) - simplified overview.

...are estimated. In addition, for C\textsuperscript{d}4 an HW\textsubscript{v} (virtual variant) is associated, this is due to a requirement on performance.

2. Wind Turbine Component Required Properties Identification and Prioritization. Based on the WTA requirements and project constraints, the $P_{IA}$ and $P_{IP}$ sets were defined. Examples of identified required properties belonging to the $P_{IA}$ sets are: (i) Max Execution Time, (ii) Required Accuracy. Whereas, examples of identified required properties belonging to the $P_{IP}$ set are: Design, Implementation, Testing, Maintenance Effort, etc. Based on design/project teams expertise, the related value are assigned. With respect to the prioritization of the required properties, different weight values are assigned to the required properties for each scenario.

3. Filtering Component Variants. In the next activities the irrelevant variants are not taken into account as follows: C1.1, C2.1, C3.2, C4.2 for the Performance-driven scenario, since they do not satisfy the required properties with respect to the max execution time, whereas C1.3, C4.4 for the Effort-driven scenario, since they do not satisfy the project constraints with respect to the development effort.
4. Defining values of exhibited properties. The exhibited properties which do not have associated values are estimated/calculated. The estimation of the is supported by the application’ simulations on the modelled plant and the estimations provided by The MathWorks Embedded Coder Toolbox used for automatic C code generation (e.g. lines of code, execution time for SW variants), HDL Coder Toolboxes for automatic VHDL code generation and by the Xilinx ISE Simulator for execution time. The estimated values are used for both scenarios. For existing components, some design effort is needed anyway to get them adapted.

5. Multiple Criteria-based Component Partitioning. Based on the information available on (i) the DM; (ii) the criteria derived by overall application requirements and project constraints; (iii) design/project team expertise; and by carrying out a visual analysis of the DM the partitioning decisions are taken as follows: for the Performance-driven scenario, the Filter, Sensor Interface and Park and Brake Controller components are deployed as hardware, while the remaining components as SW. The need of satisfying the execution time is the main driver of these partitioning decisions. For the Effort-driven scenario, the Filter, Sensor Interface components are deployed as HW, while the remaining components as SW. The main driver in this case, is the optimization of overall design cost of the components. Figure 5.5 shows the selected variants for both scenarios. The key difference is given by the Park and Brake Controller component deployment. It is deployed as an HW component in the Performance-driven scenario and as a SW component in the Effort-driven
5.5 Related Work

Over the last decades several application architectural partitioning approaches and procedures mostly oriented towards solutions satisfying performance were proposed, e.g. [9], [10]. Comparisons of the most well-known approaches are cited in [11]. Over the time the increase in complexity required new partitioning approaches, which were focused only on few low-level extra-functional requirements: combination of design costs, energy consumptions, performance as discussed in [12], but still no scalable for handling a large number of requirements. In difference to these approaches, we provide a wider-spectrum method able of considering a larger number of requirements and constraints derived by the application and, here new, by the project. In [13], a general approach for performing quantitative analysis of architectural designs based on a well-defined criteria is proposed. The approach enables to quantitatively rate design architectural alternatives based on performance metrics. In particular, the so called Y-chart approach is presented and further discussed in [14], which identifies the three core main elements influencing the choices in finding feasible solutions. It is mainly focused on performance analysis for a given set of applications, i.e. video-signal processing applications, and further work is left to different domains.

5.6 Conclusions

In this paper we have presented a method MULTIPAR for a systematic and sustainable decision process for partitioning embedded component-based systems into HW and SW. We have presented a metamodel suitable for enabling the partitioning, the method foundation and its process flow.

The novel parts in the method are (i) specification of embedded systems with components as HW and SW implementations, which required some adaptations of the component-based principles, (ii) the model-based process, starting at platform independent level and a offering solution for technology selection enabling high level component reuse, (iii) inclusion of application requirements and project constraints in the partitioning decision process.

For the future work we plan to refine the process, related to the above mentioned challenges, and then to improve the particular activities, in particular
choosing the most appropriate MCDA method. Finally our intention is to provide an integrated tool support and evaluate that in a larger industrial context (e.g. in a development of wind turbine control systems in a product line), assuming reuse of the existing components during a larger period.

References


Chapter 6

Paper II: Architectural Decisions for HW/SW Partitioning Based on Multiple Extra-Functional Properties

G. Sapienza, I. Crnkovic and P. Potena.
Abstract

Growing advances in hardware technologies are enabling significant improvements in application performance by the deployment of components to dedicated executable units. This is particularly valid for Cyber Physical Systems in which the applications are partitioned in HW and SW execution units. The growing complexity of such systems, and increasing requirements, both project- and product-related, makes the partitioning decision process complex. Although different approaches to this decision process have been proposed during recent decades, they lack the ability to provide relevant decisions based on a larger number of requirements and project/business constraints. A sound approach to this problem is taking into account all relevant requirements and constraints and their relations to the properties of the components deployed either as HW or SW units. A typical approach for managing a large number of criteria is a multi-criteria decision analysis. This, in its turn, requires uniform definitions of component properties and their realization in respect to their HW/SW deployment. The aim of this paper is twofold: a) to provide an architectural metamodel of component-based applications with specifications of their properties with respect to their partitioning, and b) to categorize component properties in relation to HW/SW deployment. The metamodel enables the transition of system requirements to system and component properties. The categorization provides support for architectural decisions. It is demonstrated through a property guideline for the partitioning of the System Automation and Control domain. The guideline is based on interviews with practitioners and researchers, the experts in this domain.
6.1 Introduction

In recent years, diverse hardware technologies have enabled a significant improvement in software performance. These hardware technologies offer heterogeneous platforms consisting of different computational units on which a particular application utilizes the specific properties of the platform. In addition to the already-present multicore CPUs, other computational units such as GPUs (Graphical Process Unit) and FPGA (Field-Programmable Gate Array) are becoming available for general-purpose software applications. This capability introduces software into new domains, and enables more sophisticated applications, but it also poses new challenges for software development. Although the computational units are characterized by particular features (such as full parallelism, or fast process context switch) it is not always obvious which parts of a software application should be deployed on which unit. This is especially true for different types of embedded systems, or cyber-physical systems (CPSs), which have specific requirements of runtime properties such as performance, resource consumption, timing properties, dependability, and lifecycle properties such as productions costs. In particular, the architectural decision about HW/SW partitioning, i.e. which application components will be implemented and deployed as software executable units (e.g. the compiled C/C++ source code), and which as hardware executable units (e.g. synthesized from VHDL), is becoming increasingly challenging. The partitioning decision is a known problem and there is a considerable body of knowledge related to that (e.g., [1], [2], [3]). In these approaches, a few factors are typically taken into consideration for a trade-off partitioning decision: e.g. resource availability (power, CPU utilization) and performance. However, due to the increased complexity and demands on system and project performance efficiency, partitioning decisions are related to many requirements, not only to run-time properties, but also to project constraints (such as available expertise, or development costs), or to business goals (such as development of mass-products, or product-line, etc.). This makes the design process quite complex and inaccurate in taking ad-hoc decisions, or manually processing all requirements. While many such decisions depend on the architect’s expertise and gut feeling, it is not guaranteed that a good (not to say the best) decision can be taken. To be able to come to an accurate decision, we must take a systematic and, when possible, an automatic approach to provide the decision.

In our previous work [4] and [5] we proposed a partitioning decision process, MULTIPAR, for component-based CPSs based on a) transformation of the requirements and constraints to Extra-Functional Properties (EFPs) through
Software Architecture, and b) Multi-Criteria Decision Analysis (MCDA) of component EFPs that depends on the component implementation and deployment (as HW or SW units). MULTIPAR enables the consideration of many component EFPs identified in the architecting process, and the discovery of a (semi)optimal deployment architecture in respect to the HW/SW deployment. This approach is appealing since it takes into consideration many requirements and many properties that reflect not only run-time aspects but also business and development project-related aspects. It does, however, introduce a more complex decision process. To make such a process feasible, two important questions must be addressed:

1. *How to specify different EFPs in a uniform way so that it is possible to use them in an deployment analysis?*

2. *Which EFPs depend, and to which extent they depend on the deployment?*

The goal of this paper is twofold: a) to provide a theoretical model of component-based systems with HW/SW components and their properties, and b) to categorize component properties in respect to HW/SW deployment. The first part includes the extension of some existing component models. The extension is necessary because of the dual nature of component implementations, namely as HW or SW units. The second part includes an analysis of EFPs defined in several standards and quality models, in respect to HW/SW deployment, and then a discussion with researchers and industry experts in the Automation and Control domain, which results in a comprehensive survey of EFPs and the impact of the partitioning decisions on their values. The concrete contribution of this paper is a) a component and EFP model for HW/SW component-based systems, b) categorization of EFPs in respect to partitioning, and c) analysis of the impact of HW/SW partitioning on EFPs for the Control and Automation domain, provided by experts from industry and academia.

The paper is structured as follows. Section 6.2 describes a formal model of component-based CPSs including EFP specifications. In Section 6.3, the paper gives the partitioning quality framework with system architecture meta-model, together with a categorization of EFPs. Section 6.4 provides an analysis of EFPs in the System Automation and Control domain, based on an empirical survey. Related work is described in Section 6.5 and, finally, Section 6.6 concludes the paper.
6.2 Modeling HW/SW component-based systems

The main idea of MULTI PAR is to automate the HW/SW partitioning decision based on the properties of components, either implemented as SW or HW. The MULTI PAR approach uses the Model-Driven Architecture with (i) the Platform-Independent Model (PIM) stage for initially achieving technology-independent design and (ii) the Platform-Specific Model (PSM) stage for subsequently enabling HW-specific and SW-specific designs. This implies first the design of the software architecture, with the identification of components and the connection between them, and then the design of the deployment view. The focus of MULTI PAR is the partitioning decision process in the PSM architecting stage.

An equally important approach in MULTI PAR is the reuse of existing components, for which we not only have the available implementation, but also specifications of their EFPs within a particular context (e.g. execution platform, implementation, etc.). A critical part of this kind of approach is breaking the system requirements down into component requirements, and then selecting the most appropriate components, i.e. selecting the components whose properties provide the best solutions in respect to the requirements.

We extend the component-based development (CBD) technique a well-known approach in SW development but not used for development of both HW and SW - to represent HW components as well. We adapt the component-based system formalism provided in [6] in order to define (i) the system as a number of components able to represent both SW and HW components, (ii) the interconnections between the components, and (iii) the platform on which the components are deployed.

We define a system $S$ that consists of a platform $P$ and a set of applications $A$ that includes $n$ applications $A_i$, i.e.

$$S = < P, A > \text{ where } A = \{ A_i \}, \ i = 1..k \quad (6.1)$$

Note that the platform can be distributed, i.e. it can consist of different and multiple execution units. We consider applications built from new and existing components, i.e. component-based applications. A component-based application $CBA$ is formally described as a pair of the following elements:

$$CBA = < C, B > \quad (6.2)$$

where $C$ represents the set of components in which the application is decom-
posed\(^1\): \(\mathbb{B}\) represents the set of bindings interconnecting the components, referred to as connectors.

A component is meant to be a modular, deployable, reusable and replaceable self-contained unit of one or more functionalities of the application. It can be implemented and deployed as HW or SW, i.e. it can either be synthesized into HW blocks or compiled into SW executable machine code. Some examples of CPS application components are: a PI-regulator, a Robot-axis controller, a FIR filter, etc.

Each component \(C\) (later deployed as either HW or SW) is characterized by a number of properties: functional properties, specified by its interface \(I\) and extra-functional properties (EFPs):

\[
C = < I, P > 
\]  

(6.3)

Functional properties are expressed as provided interface. In addition a component has a required interface that specifies the functions the component is using. A simple example of the interface with respect to a PI-Controller component may be the error signal as required interface and the controlled output as provided interface. Examples of EFPs are: execution time, memory size, reliability, etc. For each component \(C\), which is represented as a model (i.e. a specification), there can exist more implementations, i.e. component variants.

\[
C = \{ C_i \} : i = 1..n 
\]  

(6.4)

\[
C_i = < I, P_i >, \quad P_i \subseteq P 
\]  

(6.5)

Where \(C_i\) is the i-th instance (also called a variant) associated to the component \(C\) and \(n\) is the total number of variants for that component. Each variant implements the same interface, but the EFPs, defined as \(P_i\) may vary from variant to variant.

\[
P_i = \{ P_{ij} \} : j = 1..m 
\]  

(6.6)

where \(m\) is a number of specified properties for \(C_i\).

Note that this is significantly different to the specification in most component models (see in [6]), in which components comply to the same rules, i.e. to the same interface and the same EFPs. The different implementations of a component not only provide different values of the same properties, but may

\(^1\)If not explicitly specified differently, by “components” we assume “application components”
also have different properties. This is a direct consequence of their implementations that can be SW or HW.

**Required and Exhibited Properties.** The EFPs specified above are the properties *exhibited* by the components, i.e. they are immanent parts of the component instances. We designate these properties as $P_{iE}$. These are different from the *required properties* \[7\]. The required properties $P_{iA}$ are the required values of the component properties which are derived from the application requirements, the project constraints, and the architectural analysis. The application requirements typically consider run-time aspects of the application, and the required properties $P_{iA}$ derived from the application requirements define the values related to the run-time component properties. Examples of $P_{iA}$ are component execution time, and memory size that can be allocated by the component. The project constraints are usually related to the product lifecycle and business issues, and $P_{iP}$ represents the required properties derived from these project constraints. For instance, $P_{iP}$ related to the efforts and costs are the time estimated for components adaptation, design, and testing, or the cost needed for purchasing Intellectual Properties (IP) components.

For a particular application, not all available component properties are of interest, but only a subset that includes those properties that are identified from the application requirements and project constraints. Additionally, a component instance may not have all properties specified that are defined as required properties. In this case the architect/expert has to provide the values of these properties, either by estimation, by measurement or from historical data. To achieve correct design partitioning, a set of all the relevant exhibited properties $P_{iE}$ must satisfy the required properties $P_{iA}$ and $P_{iP}$ for each component instance $C_i$.

$$P_{iE} \models \{P_{iA}, P_{iP}\} \quad (6.7)$$

The condition (6.7) is required but it does not necessarily provide an optimal solution. Many configurations from different instances can satisfy this rule. To find an optimal solution, a cost function that includes the weighted sum of the properties must be defined. There exist different methods to provide this result, one of which is a Multi-Criteria Decision Analysis (MCDA).
6.3 The partitioning quality framework

The next steps in our formalization are i) specification of a property metamodel which is able to encompass the heterogeneity derived by the intrinsically different nature of HW or SW components, and ii) categorization of EFPs with respect to their sensitivity to the HW/SW partitioning decisions.

6.3.1 Component property metamodel

The property metamodel is a part of the entire component-based system metamodel, which is based on the definitions from the previous section. Figure 6.1 shows a simplified overview of the component-based system; a completed metamodel and detailed description can be found in [5]. Note that here, in contrast to typical metamodels for software components, different variants of the same component can have different EFPs. A component variant type is related to the component implementation: it can be SW, HW, or “virtual”; the “virtual” type is defined for not yet implemented components.

Figure 6.1: Component-based System Metamodel (Simplified Overview).

Figure 6.2 shows an overview of the diagram describing the Component Extra-Functional Property. The key entities of this metamodel are: the Component Extra-Functional Property, the ComponentVariantPropertyValue and the
ComponentPropertyCategory.

Figure 6.2: Component Extra-Functional Property Metamodel.

Component Extra-Functional Property. The central core is represented by the component EFP entity which abstracts a component property. It belongs to a specific category as modeled by the ComponentPropertyCategory and has an associated set of values related to a specific component variant.

ComponentPropertyCategory. There exist several different categories of properties. This categorization is used to group similar properties. The categories are listed through the ComponentPropertyCategories as shown in Figure 6.2. Each category can also be divided into subcategories. This enables categorizations that follow different standards, or it can be used for grouping complex properties that are related to a number of other (sub)properties - for example a performance, or a categorization used in a particular domain to group strongly
related properties, for instance, the COST subcategory can include costs for the design, implementation, verification and validation, and maintenance. The definition of the subcategory refers to the value associated to a specific component variant.

**ComponentVariantPropertyValue.** The ComponentVariantPropertyValue abstracts the value associated with the property. Each value has associated (i) a metric to quantitatively or qualitatively express the value (or set of), and defined through a description, a format (described by the FormatType entity), and (ii) a context which serves to capture the conditions under which the value is estimated, simulated or measured (as described by the ObtainedAsType).

### 6.3.2 The categorization of partitioning-related properties

The main driving question in this research is *which are the properties whose values strongly depend on the partitioning decisions, and which properties are independent of the partitioning?* To answer this question we provide (i) a list of possible EFPs and their categorization following the metamodel shown in Figure 6.2, and (ii) an analysis of the EFPs in respect to their dependencies on the partitioning.

To provide a list of all possible EFPs we use three types of sources a) existing quality models and standards; b) existing literature related to the partitioning decisions; and c) experience from the practice provided by experts in the field.

- **Quality models and standards.** Specifically, we exploited the following standards and quality models.

  1. ISO/IEC International Standard Software product Quality Requirements and Evaluation (SQuaRE) [8] defined for software qualities, i.e. an extension of the International Standard ISO/IEC 9126 [9], which was extensively exploited in the state of the art;
  2. Several well-known quality classifications, in particular the McCall’s quality model [10], the Boehm’s quality model [11], the Laprie’s dependability tree (attributes) [12];
  3. The quality attribute utility tree modified for embedded hardware platforms (later referred to as HW-ATAM) [13] based on the well-known Architecture Tradeoff Analysis Method (ATAM-CYC [14]);
• **State Of The Art (SOTA).** There is a rich body of knowledge related to HW/SW partitioning. We have extracted the EFPs addressed in these works. The details of this work are given in Section 5 (Related Work).

• **State Of The Practice (SOPA).** We provided a survey carried out within the iFEST (industrial Framework for Embedded Systems Tools) Artemis JU research project, with a consortium of more than 20 companies and universities in which discussion was held with the project members about important EFPs and their relation to the HW/SW partitioning. To this end, we conducted an empirical study that, in combination with the survey, included data collection through in-depth interviews with experts and researchers from both companies involved in the iFEST project, and companies and universities in Sweden. The details of this study are presented in Section 4.

We categorized the EFPs in three main categories: i) LifeCycle EFPs; ii) RunTime EFPs; and iii) Project/Business-related EFPs. As presented in [7], the system lifecycle perspective encompasses those properties which are related to the system development process and its maintenance, while the runtime perspective interests those properties whose behavior is visible, applicable, and measurable at run-time. In summary, all identified and of-interest properties are divided into three categories, namely “LifeCycle”, “RunTime” and “Project/Business-related”.

The table shown in Figure 6.3 reports the details of our categorization. The table is structured by category i.e. it is divided into “LifeCycle”, “RunTime” and “Project/Business-related”. Each category is then divided into subcategories, which group properties having similar descriptions. This is done in order to improve understandability of the categorization. However, as in most of the cases in the table shown in Figure 6.3, this does not preclude a subcategory from also being a property. In particular, in this table each category is organized as follows: the first and second columns represent the subcategories and the group of related properties, respectively. For instance, in the first row of the LifeCycle category, Usability is a subcategory grouping properties such as Appropriateness, Recognizability, User error protection, User interface aesthetics, Learnability and Training. In addition to the subcategory name, the sources from which the property is taken are reported. The categorization follows the classification in the standards and quality models (SQuaRE, ISO9126, McCall, etc.) as much as possible. Using the example just mentioned above,
for the Usability subcategory we have to exploit both the SQuaRE and the McCall’s quality model in order to identify its related properties. For each property (i.e., related to a given subcategory), its source is also reported in parenthesis.

We can observe that LifeCycle properties are present in the largest quantity. Short descriptions (titles) of the properties can be found in [15]. Many of those properties are difficult to measure. The RunTime properties are extensively studied in research and used in practice, and it is possible to express many of them quantitatively. The Project-related properties are time- and effort-related properties, as properties of project-related activities. The Business-related properties consider types of products (like mass products, product families, and similar). A relevant observation is that some properties have subcategories in different categories. For instance, Reliability is a RunTime property, but many LifeCycle properties have direct impact on the reliability. Such properties are marked in the table with an “*”.

6.4 Survey of EFPs in the Automation and Control Domain

The table shown in Figure 6.3 presents a list of possible EFP candidates which may be important in the partitioning decision process. The influence of partitioning on specific EFP also depends on other factors, for example on particular architectural solutions, or the overall goals of the application, so it is not possible to provide a general list valid for any type of application. However, in specific domains in which there are similar architectural solutions, and similar non-functional requirements, it is more feasible to provide such a list. We aim to identify which EFPs are of interest for partitioning in the Automation and Control domain, which covers a vast range of business domains, such as energy generation and transportation, industrial plants (chemical, pulp and paper, food industry), medical instruments, etc.

In order to achieve this goal we conducted an interview-survey involving several companies working in the Automation and Control domain. We specifically targeted companies whose products vary from low voltage systems (e.g. motor controllers, industrial robots) up to systems for controlling electric power transmission lines. The interview-survey research followed a method specified in [16]. An overview of data collected is presented here. Complete documentation of the results can be found in [15].
### LifeCycle Category

<table>
<thead>
<tr>
<th>Subcategory</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usability</td>
<td>Recognizability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Correctness (McCall)</td>
</tr>
<tr>
<td></td>
<td>Compatibility (McCall)</td>
</tr>
<tr>
<td></td>
<td>Consistency (McCall)</td>
</tr>
<tr>
<td></td>
<td>Reliability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Portability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Access Control (McCall)</td>
</tr>
<tr>
<td></td>
<td>Reliability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Security (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Integration (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Portability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Accessibility (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Compatibility (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Interoperability (McCall)</td>
</tr>
<tr>
<td></td>
<td>Co-existence (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Interoperability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Compatibility (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Human Engineering (Boehm)</td>
</tr>
<tr>
<td></td>
<td>Human Engineering (Boehm)</td>
</tr>
<tr>
<td></td>
<td>Communication (Boehm)</td>
</tr>
</tbody>
</table>

### RunTime Category

<table>
<thead>
<tr>
<th>Subcategory</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usability</td>
<td>Recognizability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Correctness (McCall)</td>
</tr>
<tr>
<td></td>
<td>Compatibility (McCall)</td>
</tr>
<tr>
<td></td>
<td>Consistency (McCall)</td>
</tr>
<tr>
<td></td>
<td>Reliability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Portability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Access Control (McCall)</td>
</tr>
<tr>
<td></td>
<td>Reliability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Security (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Integration (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Portability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Accessibility (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Compatibility (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Interoperability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Human Engineering (Boehm)</td>
</tr>
<tr>
<td></td>
<td>Human Engineering (Boehm)</td>
</tr>
<tr>
<td></td>
<td>Communication (Boehm)</td>
</tr>
</tbody>
</table>

### Project/Business-related Category

<table>
<thead>
<tr>
<th>Subcategory</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintainability</td>
<td>Maintenance (SQaRE, McCall, Laprie)</td>
</tr>
<tr>
<td></td>
<td>Analyzability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Reusability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Concurrency (McCall)</td>
</tr>
<tr>
<td></td>
<td>Modifiability (SQaRE)</td>
</tr>
<tr>
<td>Cost</td>
<td>Design Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Implementation Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Integration Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Testing Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Production Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Maintenance Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Development Environment Cost (SOPA)</td>
</tr>
<tr>
<td></td>
<td>Maintainability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Functional Suitability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Functional Appropriateness (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Functional Completeness (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Functional Correctness (SQaRE)</td>
</tr>
<tr>
<td>Performance</td>
<td>Performance Efficiency (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Resource Utilization (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Time Behavior (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Capacity (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Power Consumption (HW-ATAM, SOPA)</td>
</tr>
<tr>
<td></td>
<td>Schedulability (SQaRE, SOPA)</td>
</tr>
<tr>
<td></td>
<td>Reliability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Availability (SQaRE)</td>
</tr>
<tr>
<td></td>
<td>Accessibility (SQaRE)</td>
</tr>
</tbody>
</table>

Figure 6.3: Tables of LifeCycle Category, RunTime Category, Project/Business-related Category.
6.4.1 Objective

To identify the key partitioning properties for the aforementioned domain, for each EFP from Figure 6.3 we pose the following questions:

RQ1 Does the partitioning choice (HW or SW) have a direct impact on the property?

RQ1 To ensure or achieve this property would you prefer to have a solution in HW or SW or it Does Not Matter?

With these two research questions, we aim to (i) identify the EFPs which have the most impact on the partitioning decisions (related to RQ1), and (ii) provide guidance for architects (related to RQ1 and RQ2) by highlighting a default preference (HW or SW) for the realization of each property.

6.4.2 Survey Design and Process

The interview-survey was carried out among 15 experts. The participants were selected based on their affiliation and expertise. The participants were selected from industry (in total five companies and an industrial research center) and from academia (two universities). All participants had more than five years of experience as system architects (i.e., experience with both HW and SW design) and nine of them had been working for more than 15 years in the field. In addition, we selected the participants according to their prevailing work experience, with the participants pool balanced as follows: five specialists with prevalent work experience in HW design, five specialists in SW design, and five specialists in both HW and SW design. Table 6.1 summarizes the distribution of the participants with respect to their work experience (i.e. work specialization, affiliation and years of experience in the field).

The interaction with the participants was organized in two phases: 1) an introduction to the interview-survey, which was arranged in the form of individual face-to-face meetings. The purpose of the meetings was to get the participant familiar with the overall goal of the research, presenting the EFP categorizations, running through the EFP lists, and finally explaining how to fill out the questionnaire; and 2) the completion of the questionnaire by the participants.

The questionnaire consists of three main parts, related to the three main EFP categories. For each EFP specified in the table in Figure 6.3 the participants answered RQ1 and RQ2, as shown in Table 6.2. The complete questionnaire is presented in [15].
### 6.4 Survey of EFPs in the Automation and Control Domain

#### Table 6.1: Distribution among the participants

<table>
<thead>
<tr>
<th>No</th>
<th>Profession</th>
<th>Affiliation</th>
<th>Expert</th>
<th>Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>System Architect</td>
<td>Industry</td>
<td>HW</td>
<td>15 years</td>
</tr>
<tr>
<td>2</td>
<td>System Architect</td>
<td>Industry</td>
<td>HW</td>
<td>15 years</td>
</tr>
<tr>
<td>3</td>
<td>Senior Developer</td>
<td>Industry</td>
<td>HW</td>
<td>5 years</td>
</tr>
<tr>
<td>4</td>
<td>Principal Industrial Scientist</td>
<td>Industry/University</td>
<td>HW</td>
<td>15 years</td>
</tr>
<tr>
<td>5</td>
<td>Principal Industrial Scientist</td>
<td>Industry/University</td>
<td>HW</td>
<td>15 years</td>
</tr>
<tr>
<td>6</td>
<td>Senior System Architect</td>
<td>Industry</td>
<td>SW</td>
<td>15 years</td>
</tr>
<tr>
<td>7</td>
<td>Industrial Scientist</td>
<td>Industry/University</td>
<td>SW</td>
<td>15 years</td>
</tr>
<tr>
<td>8</td>
<td>Researcher</td>
<td>Industry/University</td>
<td>SW</td>
<td>5 years</td>
</tr>
<tr>
<td>9</td>
<td>Professor</td>
<td>Industry/University</td>
<td>SW</td>
<td>15 years</td>
</tr>
<tr>
<td>10</td>
<td>Researcher</td>
<td>University</td>
<td>SW</td>
<td>5 years</td>
</tr>
<tr>
<td>11</td>
<td>Senior Product Manager</td>
<td>Industry</td>
<td>HW/SW</td>
<td>15 years</td>
</tr>
<tr>
<td>12</td>
<td>Senior Developer</td>
<td>Industry</td>
<td>HW/SW</td>
<td>5 years</td>
</tr>
<tr>
<td>13</td>
<td>Industrial Scientist</td>
<td>Industry/University</td>
<td>HW/SW</td>
<td>5 years</td>
</tr>
<tr>
<td>14</td>
<td>Senior Researcher</td>
<td>University</td>
<td>HW/SW</td>
<td>5 years</td>
</tr>
<tr>
<td>15</td>
<td>Professor</td>
<td>University</td>
<td>HW/SW</td>
<td>5 years</td>
</tr>
</tbody>
</table>

#### Table 6.2: Example of questionnaire records for the LifeCycle Category

**RQ1.** Does the partition choice (SW or HW) have a direct impact on the property?

**RQ2.** To ensure/achieve this property would you prefer to have a solution in HW or SW or it does not matter?

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Property Description</th>
<th>RQ1</th>
<th>RQ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Audit</td>
<td>The ease with which the component itself and data can be checked for compliance with standards or other requirements (McCall).</td>
<td>YES</td>
<td>HW</td>
</tr>
<tr>
<td>Accountability</td>
<td>The degree to which the actions of an entity can be traced uniquely to the entity (SQuaRE).</td>
<td>YES</td>
<td>HW</td>
</tr>
<tr>
<td>Structuredness</td>
<td>The degree to which a component possesses a definite pattern of organization of its interdependent parts (Boehm)</td>
<td>YES</td>
<td>HW</td>
</tr>
</tbody>
</table>
6.4.3 Results

The collected data is presented in Figures 6.4, 6.5, 6.6, which summarize the results related to each category respectively, i.e., LifeCycle (Figure 6.4), RunTime (Figure 6.5) and Project/Business-related (Figure 6.6 EFPs). Each figure contains two stacked bar graphs, the one on left showing the RQ1 results and the one on the right reporting the RQ2 results. The properties in the graphs are sorted in descending order, according to the impact on a given property (provided by RQ1), as judged by the participants. For each graph the vertical axis label reports the properties per category. There are 44 properties for the LifeCycle category, 31 properties for the RunTime category and 24 for the Project/Business-related category.

The graphs show percentage distributions of the answers (YES or NO) for the “Partitioning Impact” graph, and HW, SW or DNM (Does Not Matter) for “Preference (HW-SW)” graph.

6.4.4 Analysis

To start analyzing the results, we introduce two indices:

The Impact Partitioning Factor (IPF), which gives a statistical measure of the impact that a given property has on the partitioning decisions for the specific domain. This index is used to analyze the results with respect to RQ1. It is calculated by the ratio in percentage between the total number of positive answers and the number of participants. We interpret the IPF according to the Fleiss Kappa agreement interpretation\(^3\), (60-100 % - a substantial or almost perfect agreement).

The Preferable Deployment Choice (PDC), which gives a statistical measure of the deployment preference for each possibility, i.e. HW, SW or DNM (“Does Not Matter”) with respect to the given property for this specific domain.

<table>
<thead>
<tr>
<th>Impact Partitioning Factor</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-40% YES, (60-100% NO)</td>
<td>No Impact on Partitioning</td>
</tr>
<tr>
<td>40-60% YES, (30-70% NO)</td>
<td>Unclear Impact</td>
</tr>
<tr>
<td>60-100% YES, (0-40% NO)</td>
<td>Clear Impact on Partitioning</td>
</tr>
</tbody>
</table>

\(^3\)Fleiss Kappa - http://en.wikipedia.org/wiki/Fleiss'\_kappa
6.4 Survey of EFPs in the Automation and Control Domain

This index is used to analyze the results with respect to RQ2. In this case we also use the Fleiss Kappa interpretation.

**LifeCycle Category.** Adaptability, Flexibility, and Maintainability result in an “almost perfect IPF” as it can be observed in Figure 6.4, (above 80%), which we interpret as a clear impact of the partitioning on these properties. The properties such as Installability, Analyzability, Expandibility, Platform Independence, Portability, Access control, Augmentability, Integrability, Modifiability, and Upgradability have a “substantial IPF” (above 61%), so a substantial agreement that these properties are clearly affected by the partitioning decision. Conversely, the properties Accountability, Communication Commonality, Consistency, User Error Protection, Completeness, Data Commonality, Self-descriptiveness, Simplicity, Training, Understandability, Access Audit, Maturity, Learnability, Legibility, Structuredness, Traceability, Appropriateness, Appropriateness Recognizability, Usability, Correctness, Human Engineering, and Communicativeness belong to a set of EFPs that are not sensitive to the partitioning decision. With regards to RQ2 (i.e. a preferable solution) we can point out that the preferable solutions for the EFPs with a clear partitioning impact are software implementations.

**RunTime Category.** In this category the Power Consumption property is the only one that shows a clear impact of the partitioning on these type of properties. Nevertheless, properties like Co-existence, Performance, Efficiency, Recoverability, Time Behavior, Reliability, and Security show a substantial agreement in the understanding of a clear impact of the partitioning on these properties. In contrast to the LifeCycle properties, here the hardware solutions prevail to achieve or ensure these properties in an advantageous way. The properties Functional Correctness, Accessibility, Authenticity, Functional Appropriateness, Functional Completeness, Operability, Functional Suitability, and Non-repudiation are not directly affected by partitioning.

**Project/Business-related Category.** By looking at the results in Figure 6.6 we can conclude that the partitioning decisions in the vast majority have an impact on the Business- and Project-related EFPs (that are directly related to the project constraints and business goals). In the majority, the software solutions prevail. There is large number of properties here which have either an “almost perfect IPF” (6 EFPs of 24 properties in total, compared to 3 of 44 in the LifeCycle category and 1 of 31 the RunTime category) or a “substantial IPF” (10 EFPs of 24). With respect to the “almost perfect IPF”, it is interesting to note that the Maintenance-related properties (i.e. cost and lead time) have a clear impact similar to the Maintainability property in the LifeCycle category, as well as a preference for a SW implementation solution.
General remarks. General remarks in relation to the current state of the art are as follows: (i) with respect to the RunTime EFPs, the results show a confirmation of the common properties used in literature, and some new properties (such as Recoverability); and (ii) with respect to LifeCycle and Project/Business-related EFPs, it is observed that many new properties are considered important for the partitioning decision process. In particular, the outcomes confirm our hypothesis that project and business constraints should be considered in partitioning decisions, which is in general missing in the state of the art and practice today.

6.4.5 Validity

In this section we address the validity of our empirical study. To this end, we discuss the following main types of validity threats proposed in [17].

Construct Validity. As common practice, in order to assure a high construct validity (i.e., a high relation between the theory behind our study and its observation), we used indirect measures (such as the reported working hours for the questionnaire review taken as a measure of effort) to conduct all steps of our work. The steps span from questionnaire preparation through recruitment of respondents to evaluation of results. Measures have been used in order to, for example: (1) avoid mono-operation bias by selecting participants with different backgrounds and work experience (see Section 6.4.2); and (2) assure rigorous planning of the study with a solid protocol for data collection and analysis. Additional threats to construct validity are represented by, for example: (1) the questionnaire structure that facilitates the data collection; (2) the individual face-to-face meetings, based on a presentation illustrating the key concepts leading the research, to get the participant familiar with the overall goal of the study; and (3) the anonymity and confidentiality guaranteed in the processing of the results to avoid evaluation apprehension.

Internal Validity. As done in [18], we addressed the confounding variables representing a major potential source of bias in empirical studies “by exclusion and randomization”. Exclusion concerns the fact that we did not select participants who were not sufficiently experienced in HW or SW design. On the one hand, we carefully balanced our pool of participants (see Section 6.4.2) according to their prevailing work experience and affiliation. On the other hand, we have used a random sample of the population in order to avoid the well-known problem of selection bias [19]. Moreover, we addressed the issue of ambiguously and poorly-worded questions [18] as described in the following points.
Figure 6.4: LifeCycle Property Graphs. Partitioning Impact (left). HW/SW Deployment Preference (right).
Figure 6.5: RunTime Property Graphs. Partitioning Impact (left). HW/SW Deployment Preference (right)
6.4 Survey of EFPs in the Automation and Control Domain

Figure 6.6: Project/Business-related Property Graphs. Partitioning Impact (left). HW/SW Deployment Preference (right)
(1) Before being released to the participants the questionnaire was iteratively reviewed by university and industry experts. Moreover, our questionnaire was based on well-assessed standards and quality models (see Section 6.3.2). (2) To let the participants understand the background and objective of this work, we performed individual face-to-face meetings. In addition, we piloted the respondents’ work in different interactions.

**External Validity.** This validity deals with the generalization of the results outside the scope of the study. Based on discussions with the experts, our assumption is that the results are applicable to the Automation and Control domain, i.e. to a population having our adopted sampling profile. It was not the intention of this paper to deal with different application domains, but we intend to work on this aspect by analyzing possible changes in the results when domain features are adjusted.

**Conclusion Validity.** This validity is focused on how sure we can be of drawing correct conclusions about the relation between the treatment and the actual outcome we observed. We achieved reliable results by piloting the questionnaire in multiple interactions, and by providing the participants with key concepts on the study. We also assured a reliable treatment implementation by using the same treatment/procedure with all participants (e.g., we distributed the same questionnaire). We found the right heterogeneity among the respondents (i.e., we carefully defined our pool of participants by not using a very diverse group of respondents) by selecting the participants from a population general enough to not reduce the external validity.

### 6.5 Related Work

The work related to our research can be divided into three categories: (i) partitioning and HW/SW co-design; (ii) analysis of EFSPs for embedded systems; and (iii) design space exploration.

**Partitioning and HW/SW co-design.** During the last few decades, several approaches have been proposed to application architecture partitioning and procedures oriented towards solutions which satisfy performance requirements, e.g. [20], [3]. A wide range of approaches have been proposed in order to automate/support the hardware/software partitioning activity using different strategies such as dynamic programming [21], heuristic algorithm based on the tabu search techniques [22], integer programming [23], and genetic algorithms [24]. A list that categorizes these approaches as well as other approaches focused on implementation issues, can be found in [3] and [1]. However, all these approa-
ches basically provide guidelines to deciding which parts of the specification should be implemented in software and which in hardware. They do this by considering platform-related indicators, such as potential speedups, area, communication overheads, locality and regularity of computations [25]. Usually, they deal with a few EFPs and are focused only on technical issues. However, some partitioning approaches do exist which optimize the combination of extra-functional requirements (e.g. design costs, energy consumptions, performance, etc.) [26], [27], but are still limited in their total number of properties. In order to provide an answer to (i) the increase in complexity of the CPSs and (ii) the advances in underlying hardware technologies when it comes to the architectural partitioning decision, in contrast to existing approaches to a component-based application we propose a general and systematic methodology capable of accounting many properties and based on MCDA techniques which are, to the best of our knowledge, non-existent today.

Analysis of EFPs for embedded systems. In the last few years, several research efforts have been devoted to the definition of methods and tools able to predict and evaluate the quality of embedded systems (e.g., [28], [29], and [30]). In particular, different techniques have been introduced to support the specific features of an embedded system (e.g., the analysis of timing properties that are typically computational and time consuming [31], or the management, preservation and analysis reuse of EFPs that are also critical tasks [32], [33]). Research efforts have been made to support the development and adaptation of embedded systems. For example, component models have been introduced to support the development of embedded systems (e.g., [34], [35], and [36]), and approaches to the adaptation of embedded systems under EFP constraints have been proposed (see, e.g., [37], [38]). Several approaches have also been introduced to estimate EFPs for software and hardware implementations (see, for example [39], [40], [41] for power and energy estimation). The run time of a hardware implementation on a FPGA, for example in [42] is estimated by exploiting the simulation and a performance model of the FPGA. In contrast, a statistical approach is proposed in [43] to estimate the execution time of embedded software.

The topic of the definition of metamodels to specify EFPs has been also studied intensely. For example, in [44] fault tolerance aspects were covered by using a metamodel, while in [45] a service-oriented metamodel for distributed embedded real-time systems covers real-time properties of services, like response time, duration, and deadline.

Other challenges related to quality analysis are seen in the strict design constraints (typical, for example, of mission-critical systems [46]) that affect
the interaction between hardware and software components. Several papers have focused on these hardware/software co-verification issues (see for example [48] and [49]). In the recent decades, digital and software designs methodologies have become more alike [2] and, as already foreseen in [50] they require designers to have a unified view of software and hardware, which converge with the concurrent design of hardware/software [2] (see, for example, the HWSWCO project and the co-design framework in [47]).

**Design space exploration:** As recalled in [51], design space exploration is an umbrella activity defined by a set of tasks addressing aspects of representation, estimation, and exploration algorithms. An overview of design space exploration techniques, such as those used for System-on-a-Chip architectures can be found in [25]. Several frameworks and tools have been built for the efficient multi-objective exploration (e.g., energy/delay tradeoff [52], occupation of the FPGA, load of the processor, and performance of the application tradeoff [53]) of the candidate architectures in order to, for example: (i) enable run-time resource management in the context of multiple applications [52], or; (ii) simulate the target system and dynamically profile the target applications by exploiting heuristic algorithms for the reduction of the Pareto set exploration time [53].

6.6 Conclusions and future work

In this paper, we have (i) presented a theoretical component-based approach to support the deployment of CPSs (as HW or SW components) into heterogeneous platforms based on many EFPs, and (ii) categorized EFPs in relation to the HW or SW deployment, and analyzed the HW/SW component deployment impact on these EFPs in the Automation and Control domain.

Specifically, with respect to the state of the art, the novelty of our contributions can be summarized in the following points. To the best of our knowledge, this is the first paper that (i) proposes a metamodel that enables the management of both application requirements and project/business constraints for the CPS application deployment, and (ii) provides a categorization for HW/SW deployment of EFPs, derived by different quality models, standards, literature and the current state of practice.

The analysis of the categorization has led to some interesting findings: i) Although the values of EFPs depend on the application architecture and the

---

4 It is also called co-simulation in the hardware industry [47].
5 http://hwswoodesign.gmv.com/
runtime context, there are EFPs that are strongly influenced by the partitioning decision. In general HW solutions provide better runtime EFPs related to system performance, while SW solutions provide better characteristics of lifecycle EFPs, such as modifiability, evolvability, and variability. i) The project constraints play important roles in the partitioning decision process which in general lacks support in existing partitioning approaches.

Our intention with the categorization and partitioning impact analysis is to provide support to system and software architects in taking architectural deployment decisions. Additional work may be required in the specialization of the EFP list, and in the partitioning impact on the EFPs. Furthermore, the findings will be integrated into our MULTIPAR framework in the form of semi-automatic support.

References


Chapter 7

Paper III:
On Applying Multiple Criteria Decision Analysis in Embedded Systems Design

G. Sapienza, G. Brestovac, R. Grgurina and T. Seceleanu.
Abstract

We focus here on the application of Multi Criteria Decision Analysis (MCDA) techniques in hardware / software partitioning activities to be used in the design and deployment of embedded systems. Our goal is to identify the best existing methods and tools suitable to support the approach we have taken for the partitioning process. We provide this via a survey of the most well-known MCDA methods and tools (for a specific class of MCDA methods called Multi Attribute Decision Making (MADM)).

We identify a set of criteria that need to be addressed, in some way, by the methods, and implemented by related tools. These “suitability criteria” help us in deciding the appropriateness of the analysed methods and tools for the envisaged partitioning approach. In brief, we are interested that the MCDA methods are taking into account multiple extra-functional properties, expressed by a variety of types, with possible missing values, should enable dependency handling, decision traceability, etc.

The conclusion is that there are criteria that are not fulfilled by any of the methods, and hence there is no method or tool that can directly used for the partitioning. However, the results shows the potential of using MCDA in the partitioning process and provide a good starting point for future research activities.
7.1 Introduction

Recently, emerging hardware technologies (heterogeneous platforms consisting of different computational units such as FPGA, CPUs and/or GPUs) have opened new perspectives for embedding more and more sophisticated functionalities into industrial applications. If, from one side, this provides a valuable answer to satisfy the market demands and the needs of launching long-term competitive products, from the other side it significantly increases the complexity of architecting such applications with respect to their deployment into hardware (e.g., VHDL) and software (e.g., C code) executable units. In this context, the design activities challenge system architects and developers to deliver implementation solutions which are the results of architectural decisions derived from several conflicting functional and non-functional (here also referred to as extra-functional) requirements [1]. The latter include, for instance, runtime and life cycle requirements as well as project constraints and business goals. As pointed out by Falessi et al. “the architecture of a system is a set of decisions” [2] and these decisions have a crucial impact on the overall development process and product lifecycle. Consequently, “making them wrong” negatively impacts business market success and product sustainability for potentially long periods of time.

A recent and comprehensive survey [3], carried out in the automation domain, confirmed that, according to the experts, there exists a large number of extra-functional properties (EFPs) which express both application requirements and project/business-related constraints, that have a significant impact on deployment decisions. In addition to this, the survey highlights that EFPs related to the project development and business constraints clearly play a key role in these decisions.

Over the last two decades, the deployment problem has been widely discussed and tackled under the umbrella of the co-design as the partitioning problem [4] [5] [6]. Several approaches have been proposed, with the most significant ones presented in [5]. These address the problem from a runtime behavior perspective, which in concrete terms means to undertake deployment strategy based just on physical metrics like performance, memory size, power-consumption, etc. To the best of our knowledge, with exception of the MUL-TIPAR approach [7] [8], there are no available solutions today which tackle the partitioning problem from a multi decision perspectives, and, at the same time:

a) support the consideration of many - possibly conflicting - project and business-related constraints, lifecycle and runtime EFPs in form of decision criteria
b) allow to take _explicitly_ into account all stakeholders’ preferences and points of view by considering specific weight for the EFPs

c) support a participative decision making process

d) facilitate the reuse of existing computational units

In literature [9], MCDA techniques are divided into two main classes: Multi Objective Decision Making (MODM) and Multi Attribute Decision Making (MADM). Existing partitioning approaches apply MODM [10] [11] [12] on a limited number of physical metrics where “an infinite number of continuous alternatives are defined by a set of constraints on a vector of decision variables” [13]. Conversely, the MADM class can be described by the following main aspects:

a) “allows for investigation and integration of the interests and objectives of multiple actors since the input of both quantitative and qualitative information from every actor is taken into account in form of criteria and weight factors” [13]

b) captures aspects of realism [9], that is, it addresses concrete properties and, in addition, does not require a user level mathematical abstraction of the problem due to the fact that this is implicitly already encapsulated in the method itself

c) may operate with missing, imprecise, uncertain data [9]

d) concentrates on problems with a discrete number of alternatives [13]

e) systematically and explicitly specifies “how attribute information is processed in order to arrive at a choice” [14]

In the present research work, we are interested in evaluating the suitability of MCDA methods (and related tools) for supporting a partitioning process from a multi attribute decision perspective.

Eventually, we aim to find the answers to two main research questions:

1. Which are the criteria that are most relevant to assess the suitability of MCDA methods for the aforementioned aspects?

2. Which (collection of) MCDA method(s) is suitable for achieving partitioning solutions satisfying these aspects?
Hence, we want to investigate the MCDA methods and tools for the capabilities that enable or support:

a) the consideration of multiple extra-functional properties
b) the reuse of existing computational units
c) a systematic and participative decision making process

In the following, we start by defining the key requirements of interest for partitioning embedded applications into hardware and software units. Based on these, we derive “11-suitability criteria” to assess MCDA methods for the given problem. Then, we identify the MCDA methods available today and based on the “11-suitability criteria” we assess their suitability and also that of the related tools. Our main contributions are:

a) the identification of a number of criteria to assess the suitability of MCDA methods with respect to the hardware/software partitioning
b) the suitability assessment of MCDA methods for partitioning.

To address item b) above, we executed an investigation which included more than 100 methods and tools. These results are presented in the main body of the paper.

For the sake of completeness, we also include an Appendix section, which includes the references (e.g. links and other material) to all of the methods and tools that were part of the investigation.

7.2 Research Context and Related Work

We present here a number of basic MCDA principles in order to motivate our approach and forthcoming choices. With respect to the MCDA background, our main goal is to highlight the key concepts of available techniques for the MADM class which are of special interest for our research. Then, we discuss the related research work on MCDA suitability for partitioning.

7.2.1 Basic Concepts of Multiple Criteria Decision Analysis

Multiple Criteria Decision Analysis (MCDA) (also referred to as Multiple Criteria Decision Making (MCDM), or Multiple Criteria Decision Aid) is a branch of operational research which refers to “making decisions in the presence of
multiple, usually conflicting, criteria” [15]. MCDA is defined as “an umbrella term to describe a collection of formal approaches which seek to take explicit account of multiple criteria in helping individuals or groups explore decisions that matter [16]. It is widely applied in many fields like medicine [17], forestry [18], economics [19], energy management [20], transportation [21], watershed land resource allocation [22], assessing the re-manufacturability and re-usability of products, materials or components, to assess the best transportation network [23], for renewable energy planning and policy [13], or to solve other general problems [9] [24].

MCDA problems are roughly divided into two main classes [9] [25] [22] [26].

A first class includes methods dealing with problems considering a finite number of possible alternative solutions, also referred to as Multiple Attribute Decision Making (MADM) [16] [13] [14]. For this class of methods the alternatives are given explicitly from the beginning, and via a well-structured decision process these methods allow to capture/include project stakeholders’ opinions into the ranking/classification/choosing of the alternatives [27].

The second class considers problems dealing with an infinite number of alternative solutions and it is usually referred as Multiple Objective Decision Making (MODM) [13] [22] [28]. In this class of problems a set of objective functions are optimized subject to a set of constraints [13] and focuses on problems where the decision space is continuous [26].

The main advantages of MCDA solutions are presented and discussed in [13] [29]. A comparison highlighting the key difference between MODM and MADM classes can be found elsewhere [22] [30] [13], but we present a short list underlining the most relevant differences:

a) In MODM, the criteria are defined by objectives, while in MADM, the criteria are defined by attributes

b) The objectives have to be explicitly defined when applying MODM methods, while in MADM they are implicitly defined

c) Alternatives are explicit (i.e. they are known) and finite, in MADM, while in MODM, the alternatives are infinite and implicitly defined.

We present below the main keywords around both MCDA concepts. As there are no global definitions of these concepts [31], authors use them interchangeably.
7.2 Research Context and Related Work

- **Attributes** can be defined as the characteristics, qualities or performance parameters of alternatives [32]

- **An alternative** represents the item to evaluate according to its attributes to find the final decision solution (or set of solutions)

- **A Criterion** [15] is a measure of effectiveness representing the basis for evaluation.

- **Objectives** are defined [32] as “reflections of the desires of the decision maker and they indicate the direction in which the decision maker wants the organization to work”.

- **Goals** (or targets) can be understood as “apriori values or levels of aspiration” that can be “either achieved or surpassed or not exceeded” [15].

A formal definition of a typical MCDM problem is provided in [33].

Our study focuses on the first class of MCDA (i.e. MADM), and below we provide a key concepts of this class of method the Decision Matrix (DMx)).

The DMx is a table in which each criterion and alternative of the problem under study is collected. A typical DMx is shown in Figure 7.1 where the set of alternatives is represented by \( A = \{a_1, ..., a_n\} \), and the set of criteria by \( F = \{f_1, ..., f_k\} \).

<table>
<thead>
<tr>
<th>( a_i )</th>
<th>( f_1 )</th>
<th>( f_2 )</th>
<th>...</th>
<th>( f_j )</th>
<th>...</th>
<th>( f_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 )</td>
<td>( f_1(a_1) )</td>
<td>( f_2(a_1) )</td>
<td>...</td>
<td>( f_j(a_1) )</td>
<td>...</td>
<td>( f_k(a_1) )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( f_1(a_2) )</td>
<td>( f_2(a_2) )</td>
<td>...</td>
<td>( f_j(a_2) )</td>
<td>...</td>
<td>( f_k(a_2) )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>( a_t )</td>
<td>( f_1(a_t) )</td>
<td>( f_2(a_t) )</td>
<td>...</td>
<td>( f_j(a_t) )</td>
<td>...</td>
<td>( f_k(a_t) )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>( a_s )</td>
<td>( f_1(a_s) )</td>
<td>( f_2(a_s) )</td>
<td>...</td>
<td>( f_j(a_s) )</td>
<td>...</td>
<td>( f_k(a_s) )</td>
</tr>
</tbody>
</table>

Figure 7.1: A simplified example of a MADM table, commonly referred to as a Decision Matrix.

The evaluation value of the alternative \( a_i \) with respect to the \( j \)-th criterion is defined by \( f_j(a_i) \).

Let us analyze those characteristics by considering a simple example of buying a house. The alternatives are House_1, House_2, House_3, and the
attributes are the price, overall impression of location, house size and comfort. The DMx for this example is shown in Figure 7.2.

The main goal of a Decision Maker (DM) is to find the (set of) alternative(s) which best optimizes/trades-off all the criteria. The solution of a MCDA problem depends not only on the evaluation of alternative values, but it is also affected by the importance of a specific criteria with respect to others, as viewed by the DM. This is modeled by the introduction of respective weights in relation to the criteria. These weights are assigned by the DM, and may vary from DM to DM, based on their specific interests.

Hence, in our example, the DM has to assign a weight to each criteria of Figure 7.2 (price, location, size, comfort) by using an appropriate weighting method. The presented values are the result of a single DM decisions (other buyers may chose differently). Then, the DM has to set his preferences with respect to each criteria, for example, the DM may prefer to buy a house with the lowest possible price but, at the same time, may prefer the largest possible house. Thus, the DM has to take into account multiple and conflicting criteria. For instance, “Price” may conflict with “House size”, and, in addition, the criteria are measured with different units (e.g. euro vs. square meters), leading to what is usually referred to as solving decision problems dealing with incommensurable units [26] [15] [34]. Moreover, criteria might be of qualitative or quantitative nature (the price of a house can be numerically measured while the comfort rating can be subjectively described as high, medium, low), or they might be deterministic or probabilistic [34] (the price of a house can be described using deterministic values while the overall impression of location could be a random value). Uncertainty also plays a key role when making
decisions. It can be caused by different nature, e.g. as derived by subjective judgement or by unknown or incomplete (imprecise) information [34]. In such a scenario, which also includes the mixing of different criteria natures, the problems are intrinsically hard to solve.

A well-known classification of the most frequent decision-making problems with respect to the MADM is provided in [9] [35]. It mainly deals with decisions related to the choice, ranking or classification/sorting of finite alternatives. Consequently, these type of problems are grouped into the following three classes of methods.

**Choice methods.** The objective of these methods is to aid DMs in selecting a subset of alternatives within constraints, as restricted as possible. For example, using these methods the DM might choose to buy two or more houses, with a budget limit of $800,000 and a minimum size limit of 206 m². Based on these constraints, the PROMETHEE V¹ method (a MCDA method for the choice problem) has proposed House_1 and House_3 as the best possible alternatives for DM.

**Ranking methods.** This class of methods is used to aid DMs in ranking all of the alternatives from best to worst. For example, using these methods the DM will get a ranking of all houses and might choose to buy the best-ranked house. The final output from PROMETHEE II method (a MCDA method for ranking problem) is House_2, House_1 and House_3.

**Classification/Sorting methods.** The goal of this class of methods is to aid the DM in assigning alternatives to a defined class. If the classes can also be ordered, then the DM deals with what is defined as ordered sorting problems. Using SMAA-TRI (a MCDA method for the classification problem [36]) the houses are placed in two classes such good or bad, i.e. the House_1 and House_2 could be classified as a good and House_3 could be classified as a bad option.

With respect to the above classification, this research focuses on methods able to solve either ranking or classification decision problems since they could both be applied to reach one or more partitioning solutions. The first type of method can be used to rank all of the hardware and software alternatives; methods of the second type can classify an alternative either as hardware or software.

¹PROMETHEE I, II and V are part of the PROMETHEE method family. The complete overview of the family can be found in [9].
7.2.2 Partitioning and MCDA Suitability Analysis Related Work

We share here the definition of the partitioning problem stated as “finding those parts of the model best implemented in hardware and those best implemented in software” [4]. Hardware/software partitioning is considered to be one of the main challenges faced when designing embedded applications: “as it has a crucial impact both on the cost and on the overall performance of the resulted product” [37]. However, this is a non-deterministic polynomial (NP)-hard problem [38]. A rigorous mathematic formulation and analysis of the partitioning problem is provided in [39].

There is a considerable body of knowledge concerning partitioning and related issues. It was studied intensively in the 1990s and the early years of the 2000s, when the co-design basically emerged as a new discipline [5]. Over the decades, a wide range of approaches have been proposed to automate/support hardware/software partitioning using different strategies, for instance dynamic programming [40], heuristic algorithms based on the tabu search, simulated annealing, genetic algorithm techniques [41] [42] [43] [44] [45] etc., integer programming [46], multiple objective optimization techniques [10] [11], etc. An in-depth study of several partitioning approaches and related issues is provided in [44] and a walk-through of the highlights of partitioning approaches over the last two decades is found in [6] [5] [47]. All of these approaches are mainly oriented towards solutions satisfying physical performance requirements. They provide partitioning solutions whose results are based on platform-related indicators, like potential speedups, area, communication overheads, locality and regularity of computations [48]. For instance, this is the case with approaches such as Vulcan or COSYMA², which target optimization problems focusing on design constraints such as timing, resource speed-up and so forth [49].

Even though few partitioning approaches exist which optimize a combination of EFPs [50] [51], they are limited with regard to the total number of EFPs they are dealing with. Additionally, current partitioning approaches only focus on technical issues (e.g. design costs - memory size, area, etc., energy consumption, performance, etc.) and do not take into account properties related to the project development and business perspectives (e.g. legacy, resource availability, design effort, testing costs, production costs, etc.), or EFPs of key importance for embedded industrial application developments such as safety, reliability, security, maintainability, sustainability, scalability, field upgradeability and so forth.

²http://www.cs.ccu.edu.tw/~pahsiung/courses/codesign/resources/codesign-tools.html
Unlike the previous work [52] [53] [54] [5], which includes recent overviews of high-level design exploration of MCDA-related partitioning approaches –especially focused on multi-objective optimization techniques – our approach is focused on MCDA techniques which deal with a finite number of alternatives, with lack of information (uncertainty) as well as with information of a different nature (e.g. qualitative, quantitative, distributions, etc) and with incompa-
rable information. Another advantage lies in the fact that it provides a mean of systematically making decisions based on different project stakeholders’ perspectives. In addition, it also brings together project-related constraints and business goals (together with lifecycle as well as runtime non-functional requirements) [3] into the set of decision criteria. For instance, in [55] the design space exploration at system level problem is tackled by using a multi-objective pseudo boolean solvers. However, the partitioning is applied on a limited num-
ber of EFPs (specifically, for the test cases the analysis is limited to just two EFPs related to the run time behaviour: the area and the power consumption). The problem of selecting the solution is left to the DM / designers, who have to select it from a continuous spectre of possible solutions in the Pareto-front. In our approach, we take into account several EFPs, and, in addition, the DM perspectives/preferences are captured before the most suitable partitioning solution is selected or ranked among a finite number of possible hardware/software configurations.

While the authors of [56] introduce the concept of a scenario-based de-
sign space explorations, the mapping is performed by either using a multi-
objective genetic algorithm method, or using dynamic sequential oscillations search methods, or applying a combination of both. However, this leads again to the problem of selection from a continuous space of possible solutions, hence this approach is applied just to a limited number of objectives (i.e. performance, energy and cost).

A hierarchical top-down optimization methodology in order to decompose the multi-objective optimization problem into sub-problems is proposed in [57]. Nonetheless, this suffers from the exact selection issues mentioned above.

Selecting an appropriate MCDA that best suits the requirements of a specific decision-making problem is not a trivial task. Over the last few decades, a large number of MCDA methods has been proposed, but they “differ in the way the idea of multiple criteria is operationalised” [30], e.g. objective, criteria-assessing, weight computation and preference model, algorithms applied, and so forth. Hence, depending on the decision-making user context and the type of problem to be solved, some methods are more suitable than others for reaching certain goals.
Several solutions have been proposed to assess the appropriateness of MCDA methods for solving the “decision-making problem at hand”, e.g. [30] [58] in which different methods are assessed with regard to sustainability issues, or [59], where the methods are assessed for their suitability in solving seismic structural retrofitting issues. The above take a similar approach to our work here, however, they deal with totally different application domains. Hence, only a few “core” concepts (such as criteria interdependence, process transparency, etc.) can be ported to our research.

In order to reach our goals of assessing the suitability of MCDA methods for partitioning, we use the results of the approaches above as a starting point, since they provide an appropriate set of general and common (to all methods) requirements that are applicable for our analysis. Then, we extend the current state of the art by assessing several MCDA methods with respect to a wide set of partitioning-related requirements, currently not available in literature.

As an example, the applicability of MCDA for partitioning was assessed in the context of the recently introduced MULTIPAR approach which enables the partitioning of the application based on:

a) the transformation of the application requirements and project/business-related constraints into EFPs

b) the reuse of existing components

c) MADM techniques

7.3 MCDA Methods - Partitioning Suitability Criteria

We identify here the features of interest that the MCDA methods need to provide from the partitioning perspective. We start by eliciting the key requirements for the partitioning process (the “Elicitation part”), and then we derive the suitability criteria of interest to carry out the assessment (the “Extraction part”).

We used the results provided by a recent project survey\(^3\) which gathered the industrial and academic experiences related to the development and maintenance of hardware/software co-design systems (heterogeneous and multicore systems) and their relation with respect to partitioning. We also ana-

\(^3\)iFEST - http://www.artemis-ifest.
analyzed the results of the empirical study on the architectural decisions for hardware/software partitioning based on multiple EFPs conducted in [3]. This provides data collected through surveys with experts and researchers from both companies and universities in Sweden. The survey from Figueira et al. [9] and works such as [30] [58] provided additional information in deriving the criteria for our assessment. Thus, [30] provides a "list of criteria on which MCDA methods can be compared in context of a certain applications" and guidelines for choosing the most suitable MCDA method for an environmental sustainable development. Even though in [30] and [58] respectively only four and seven MCDA methods were analyzed and compared, respectively, some of the criteria used for their assessment were of general nature. Consequently, we considered these as starting point in our research.

For performing the assessment, the following activities were carried out.

**Elicitation Part.** In this activity, the requirements related to the suitability assessment are elicited. In order to reduce the inconsistencies typical of any requirements elicitation process, the requirements are iteratively reviewed by practitioners and researchers from industry and academia. We identify two groups of main requirements:

1. **high-level requirements**: a collection of statements which identify the scope boundaries of the partitioning process. We further divided them into the following categories:
   - *General process features*: e.g. reliability of the entire process, traceability of the process artifacts, etc.
   - *EFPs and alternatives handling*: aspects related to the criteria and the elements that have to be partitioned (i.e. components)
   - *Decision handling*: the requirements related to the partitioning decisions and the preferences of the stakeholders (e.g. the DMs, system architects, designers, etc.)

2. **detailed requirements**: a collection of detailed requirements and of declarative statements defining what partitioning process-related solutions have to be met. We briefly present below some examples of such requirements (additional details can be found in [60]):
   - *Requirements addressing the general process features:*
     - (i) the process shall be systematic, it has to consist of a well-defined process flow and related activities
     - (ii) it has to enable reuse of existing components
(iii) it has to be able to handle the classification and/or ranking of hardware and software units to be partitioned
(iv) it shall provide metrics to assess the quality of the overall partitioning solutions

- Requirements addressing the elements to be partitioned:
  (i) there exists a finite number of elements to allocate
  (ii) elements have to be mapped either as software or hardware executable units
  (iii) there might exist software or hardware elements that have to be reused
  (iv) each element might have several associated EFPs

- Requirements addressing the handling of extra-functional properties:
  (i) there is no limit to the number of EFPs that might be associated with an element
  (ii) there exist different types of EFP-associated values, (e.g. qualitative, quantitative, probabilistic, ranges, and unknown or incomplete values)
  (iii) prioritization among EFPs or group of EFPs shall be allowed

- Requirements addressing the decision handling:
  (i) in general, there might exist several DMs [31]. However, only one is to eventually be responsible for actually taking the decisions related to MCDA context set-up
  (ii) prioritization mechanisms shall be provided in order to handle DMs and stakeholders’ preferences [31]

Extraction Part. Based on the mentioned requirements and with the support of experts from industry and academia, we derived 11 MCDA-related criteria (further referred to within the text as the 11-suitability criteria) of fundamental importance for the partitioning. We then grouped them into three classes according to their priority (see Table 1). The criteria and their respective priorities will be used to conduct the method/tool suitability assessment (see Section 7.4).
Table 7.1: The 11-suitability criteria

<table>
<thead>
<tr>
<th>High Priority</th>
<th>Medium Priority</th>
<th>Low Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qualitative Assessment</td>
<td>Probability-based Assessment</td>
<td>Inconsistency Indication</td>
</tr>
<tr>
<td>The method shall be able to estimate the solutions based on qualitative information. For instance, a component might have an EFP (e.g. expert level of confidence) whose values might expressed as: high, medium, low</td>
<td>The method shall be able to estimate the decisions on EFPs which value can be expressed by a probability distribution, e.g. the time distribution of the service maintenance per year.</td>
<td>The methods shall provide the means to detect inconsistency between the EFP values. For instance, in the case of an EFP whose value is greater than its maximum limit, the methods should provide a means to detect this inconsistency.</td>
</tr>
<tr>
<td>Quantitative Assessment</td>
<td>Interval Information</td>
<td>Decision Traceability</td>
</tr>
<tr>
<td>The method shall be able to estimate the solutions based on quantitative information.</td>
<td>The method shall be able to estimate the decisions on EFPs whose values are expressed by intervals/ranges, e.g. the estimated line of code (LOC) might be expressed with an interval such as 200-300 LOC.</td>
<td>The methods shall provide the means to trace the decision process in order to facilitate DMs’ sensitivity analysis or partial re-iterations of the partitioning process.</td>
</tr>
<tr>
<td>Unknown Information</td>
<td>Preference Elicitation</td>
<td></td>
</tr>
<tr>
<td>The method shall be able to estimate the decisions in the case of missing information. For instance, some legacy components used in a previous product might not have all the associated information (i.e. EFP values) of interest and it would be hard and time consuming to retrieve them.</td>
<td>The methods shall allow the elicitation of the preferences from the DMs (e.g. weights and their prioritization).</td>
<td></td>
</tr>
<tr>
<td>Scalability</td>
<td>Dependency Handling</td>
<td></td>
</tr>
<tr>
<td>The methods shall not be restricted to the use of a limited number of criteria (derived by EFPs) and alternatives (i.e. components).</td>
<td>The methods shall be able to model and obtain solutions which take into account the architectural dependencies that might exist (i) between the components (alternatives) e.g. the bandwidth, and (ii) between the EFPs (criteria) (e.g. safety, reliability and so forth).</td>
<td></td>
</tr>
<tr>
<td>Subjective Judgment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The methods shall provide means for the DMs to influence the decisions, e.g. by selecting preference functions, setting thresholds, and so forth.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
7.4 Assessment of MCDA Suitability for Partitioning

We gathered information on the current MCDA methods according to the problem classification presented in Section 7.2.1, dividing them into ranking, classification and choice classes.

The results of this division are summarized in Figure 7.3, which shows which MCDA methods we identified for each class. For each method, we were also interested in determining the methods ability to handle uncertainty (U), and dependency (D) and if tools (T) exist to implement the method.

The results also show that there are 86 methods for solving ranking problems, and 20 methods for solving classification problems, 14 of which can handle missing values. Several of the methods have at least one tool implementing them, while other methods (SWING, PRAGMA, etc.) do not have any associated tools. In multiple situations (e.g. IDS, SANNA, DEFINITE, etc.), some tools are implementing a limited number of method functionalities, or are providing additional functionalities that the method does not provide. No method is able to handle or model dependencies for either criteria or alternatives. The ANP method (a generalization of AHP method) comes closest to this, by allowing the construction of interconnected clusters, creating a so called “network”. All the elements (attributes) in the network can be related in any possible way, i.e. a network can incorporate feedback and interdependence relationships within and between clusters [61]. However, the biggest limitation is “the exponential growing of the complexity due to the pairwise comparison of the criteria” [62] which makes AHP unsuitable for application in the partitioning decision processes, where a large number of criteria are considered.

Of the 86 methods we identified above, we have selected 37 methods to take into account for our assessment, of which 31 are ranking methods and 6 are classification methods.

The methods belonging to a family (e.g. the ELECTRE family) were evaluated per member. Thus, we considered each method independently (e.g. ELECTRE III and ELECTRE IV).

7.4.1 MCDA Methods vs. the 11-Suitability Criteria

Figure 7.4 shows the results of our investigation, that is, the selected methods versus the 11-suitability criteria. A few brief remarks:
### Assessment of MCDA Suitability for Partitioning

#### Legend
- **U**: Uncertainty
- **T**: Tool
- **D**: Dependency

#### MCDA Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>U</th>
<th>T</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELECTRE I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELECTRE II</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELECTRE III</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELECTRE IV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE II</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE III</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE IV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE VI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE VII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE VIII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE IX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XIII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XIV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XVI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XVII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XVIII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XIX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXIII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXIV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXVI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXVII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXVIII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXIX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE XXX</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Figure 7.3: MCDA Methods

The figure illustrates the classification, ranking, and choice problems using various MCDA methods, along with their uncertainties, tools, and dependencies. The methods are categorized into ELECTRE I to ELECTRE IV, PROMETHEE I to PROMETHEE X, and additional methods like TOPSIS, VIKOR, SMAA, and others. Each method is represented with a symbol indicating its suitability for partitioning. Further details are provided in the table format above.
• All methods are able to support quantitative data, but only few of them can handle both qualitative and quantitative data (4 ranking methods and 2 classification methods)

• More than 80% of the ranking methods are capable of allowing DM preference elicitation (they allow consideration of criteria weights); this rate is lower (about 30%) for the classification methods

• The estimation of the decisions based on probabilistic and interval data is supported by only 4 ranking methods, and just one classification method addresses these

• 50% of classification methods can provide solutions under uncertainty conditions, while only around 30% of the ranking methods handle the lack of data

• Almost all methods allow scalability in terms of criteria and alternatives. However, AHP [63] [64] [65] [66], (and ANP, since based on AHP) have some limitations when the number of criteria is increasing. For example, AHP needs to perform n(n-1)/2 pairwise comparisons. Thus, as the number of criteria increases, it becomes more difficult and time-consuming to identify the values to be assigned to the weights

• Indication of inconsistency (type, etc.) is observed by a few (3) ranking methods

• For a large number of methods (27 ranking and 6 classification) it is possible to trace back how the decisions were carried out

• The ability to consider the DM judgment (see Table 1) is offered by 18 ranking and 2 classification methods

• Modelling and handling dependencies between the criteria and between alternatives is not fulfilled by any of the investigated methods

For the ranking methods, the largest number of criteria is fulfilled by the Evidential Reasoning [34] [67] method, at 8 out of 11 criteria. For the classification methods, the SMAA-TRI [36] method covers the highest number of criteria, 6 out of 11.
### 7.4 Assessment of MCDA Suitability for Partitioning

<table>
<thead>
<tr>
<th>Suitability Assessment Criteria</th>
<th>SMART weight</th>
<th>Normalized SMART weight</th>
<th>Fulfilled Criteria</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Partial SUM</th>
<th>Weighted SUM Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability-based Assessment</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Evidence-based Assessment</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Quantitative Assessment</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Qualitative Assessment</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Unknown Information</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Scalability</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Suitability Adjustment</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Stability</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Fulfilled Criteria</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Partial SUM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Weighted SUM Score</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Legend**

- ✓: Fulfilled Criteria
- □: Partial SUM
- ✓: Weighted SUM Score

Figure 7.4: MCDA methods versus the 11-suitability criteria.
7.4.2 Suitability Assessment

In order to decide which methods are the most suitable for solving partitioning problems, we applied on the results of Section 7.4.1 the Weighed Sum Method (WSM) [68] in combination with the SMART method [69] [70].

Applying WSM requires assigning a weight to each of the 11-suitability criteria (see the rows “SMART weight” and “Normalized SMART weight” in Figure 7.4), respectively. By using SMART, we assign the weights based on the priority of the criteria (the “PRIORITY” row in Figure 7.4), and on the feedback received from the experts.

The results of the WSM method are shown in the “Weighted SUM Score” column in Figure 7.4. The EVIDENTIAL REASONING method has the highest score, above 0.7. However, seven methods exist with a score greater than 0.6 (e.g. PROMETHEE I and II, REGIME, EVAMIX, etc.) There are no classification methods with a score above 0.6. The TOMASO method is the least suitable method for partitioning (lowest score 0.27).

7.4.3 Tools

The next step is to identify what are the available software tools associated with the methods and which method’s features are implemented by the tool(s).

Our investigation results in the identification of 22 tools, where 17 implement ranking methods, 4 implement classification methods and 1 implements both classes of methods.

The mapping between the methods and the tools is provided in Figure 7.5. Some of the tools implement more than one method, for instance the SANNA tool (implementing 5 methods) or DECERNS_MCDA tool (implementing 6 methods).

We performed further an analysis of the tools with respect to the 11-suitability criteria defined in Section 7.3 and three additional criteria related to tool features such as the ability to automatically generate a report (Report Generation), the possibility of visualizing the results through graphs (Graph Visualization), and tool support via manual and/or help (Manual/Help). The results are shown in 7.6. Visual Promethee Academic, Intelligent Decision Systems (IDS) and DEFINITE fulfill the largest number of criteria (10/14) for solving ranking problems, while for classification problems the JSMAA tool is the most suitable (7/14).

We decided not to evaluate some methods (such as Vikor, EXPROM1, EXPROM2, PAIRS, QUALIFLEX, STOPROM2, PRAGMA and PRO-AFTN).
### 7.4 Assessment of MCDA Suitability for Partitioning

#### Legend
- Ranking
- Ranking and Classification
- Classification

<table>
<thead>
<tr>
<th>Methods</th>
<th>ELECTRE III</th>
<th>ELECTRE IV</th>
<th>SMAA-2</th>
<th>SMAA-O</th>
<th>ANP</th>
<th>SMAA-TRI</th>
<th>SMAA-O</th>
<th>SMAA-TRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELECTRE I &amp; IV</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMAA-2</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMAA-O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Other</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOPSIS</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE I</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROMETHEE II</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evidential Reasoning</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABEM</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAVIT</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPROM-1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UTA</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAVIDE</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAUT</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AIM</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PARES</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UTA-I</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UTA^GMS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QUALIFLEX</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REGIME</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DREIS</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPROM-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOPROM-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRIP</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UMAP</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRAGMA</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROME</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EVAMIX</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KRA</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMAA-TRI</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ELECTRE TR</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOMASO</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROAFTN</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7.5: MCDA methods versus tools.
with respect to the above-mentioned tool criteria due to the fact that the associated tools were deprecated (i.e. supported only by, for example, Disk Operating System (DOS)). In addition to this, by comparing Figure 7.4 to Figure 7.6, one can notice that, a few tools do fulfill the same criteria as the methods they implement (see IDS tool and Evidential Reasoning method). However, this is not trivial for most of the other tools (Visual Promethee Academic and PROMETHEE : 7 fulfilled criteria vs. only 6, or DEFINITE and EVAMIX: 7 vs. only 6, etc.).

<table>
<thead>
<tr>
<th>TOOLS</th>
<th>Preference Elicitation</th>
<th>Quantitative Assessment</th>
<th>Qualitative Assessment</th>
<th>Probability-Based</th>
<th>Interval Information</th>
<th>Unknown Information</th>
<th>Suitability</th>
<th>Dependability Handling</th>
<th>Inconsistency Indication</th>
<th>Decision Traceability</th>
<th>Suitability Assessment</th>
<th>Stability Assessment</th>
<th>Ranking</th>
<th>Classification</th>
<th>FUllied Criterion - Total SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELECTRE III / IV</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>5</td>
</tr>
<tr>
<td>MakeItRational</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>6</td>
</tr>
<tr>
<td>ANP SOLVER</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>5</td>
</tr>
<tr>
<td>WEB ANP SOLVER</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>5</td>
</tr>
<tr>
<td>TRANSPARENT CHOICE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>6</td>
</tr>
<tr>
<td>REIPPTYCH</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>5</td>
</tr>
<tr>
<td>SANNNA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>5</td>
</tr>
<tr>
<td>Visual Promethee Academic</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>7</td>
</tr>
<tr>
<td>Intelligent Decision System</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>10</td>
</tr>
<tr>
<td>NAIAD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>8</td>
</tr>
<tr>
<td>DECERS, MCDA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>9</td>
</tr>
<tr>
<td>PRIME Decisions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>9</td>
</tr>
<tr>
<td>WEB AIM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>9</td>
</tr>
<tr>
<td>VisualUTA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>9</td>
</tr>
<tr>
<td>DEFINITE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>10</td>
</tr>
<tr>
<td>PolyDecisions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>10</td>
</tr>
<tr>
<td>Decision Deck</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>8</td>
</tr>
<tr>
<td>SIMIA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>7</td>
</tr>
<tr>
<td>ELECTRE TRI</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>7</td>
</tr>
<tr>
<td>ROSE 2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>6</td>
</tr>
<tr>
<td>4eMka2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>6</td>
</tr>
<tr>
<td>IMAF</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>6</td>
</tr>
</tbody>
</table>

Figure 7.6: MCDA tool versus the criteria (they also include the 11-suitability criteria).
7.5 Conclusions and Future Work

We presented a suitability analysis of MCDA methods for being applied in hardware/software partitioning methodologies. The analysis is focused on the MADM class of methods and it was conducted taking into consideration (more than) 86 methods and 22 tools. To the best of our knowledge this is the first study that proposes a suitability analysis of MCDA methods, and in particular with respect to the MADM class, for the hardware/software partitioning problem. It provides a set of guidelines –the 11-suitability criteria– which can be used to assess the suitability of any MCDA method and related tool(s) in the partitioning decision process. It also ranks methods according to their respective fulfillment of the mentioned criteria. An interesting finding is that currently there are no MCDA methods able to fulfill all of the 11-suitability criteria. The criteria that has not been covered by any method or tool is the Dependency Handling.

As highlighted by experts in the field, the architectural dependencies between computational units and the dependencies between different EFPs, are of key relevance in achieving efficient partitioning solutions which guarantee the sustainability of the system over its entire lifecycle. As a consequence, we see the importance of orienting our future research towards the definition of an approach able to model these dependencies and of integrating them into the MCDA ranking for hardware/software partitioning. We consider as important an assessment of the importance of each criteria within the list of 11, with respect to the quality of the eventual method and tool ranking. Additionally, the results of this study will be used to further develop MultiPar, which partitioning approach applies MCDA techniques.

Acknowledgment

This research is supported by the Knowledge Foundation through ITS-EASY, an Industrial Research School in Embedded Software and Systems, and by the Swedish Foundation for Strategic Research through the RALF3 project.

References

114 References


### 7.6 Appendix: MCDA - Methods and Tools References

Here, we provide all of the method and tool information (e.g. links and references) used for the survey. They are divided into three categories as presented in Section 7.2.1 and for each category methods and tools are shown. All links were accessed in January 2016.

#### 7.6.1 Classification Methods and Tools

<table>
<thead>
<tr>
<th>MCDA Methods and Extensions</th>
<th>Links and References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>UTADIS I</strong></td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>DC</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>N-TOMIC</td>
<td>N-tomic: A support system for multicriteria segmentation problems</td>
</tr>
<tr>
<td>UTADIS II</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>MCDA Tools</td>
<td>Links and References</td>
</tr>
<tr>
<td>ELECTRE TRI</td>
<td><a href="http://www.lamsade.dauphine.fr/~mayag/links.html">http://www.lamsade.dauphine.fr/~mayag/links.html</a></td>
</tr>
<tr>
<td>IRIS</td>
<td><a href="http://www.lamsade.dauphine.fr/~mayag/links.html">http://www.lamsade.dauphine.fr/~mayag/links.html</a></td>
</tr>
<tr>
<td>CADABRA</td>
<td><a href="http://idss.cs.put.poznan.pl/site/151.html">http://idss.cs.put.poznan.pl/site/151.html</a></td>
</tr>
<tr>
<td>JAMM</td>
<td><a href="http://idss.cs.put.poznan.pl/site/jamm.html">http://idss.cs.put.poznan.pl/site/jamm.html</a></td>
</tr>
<tr>
<td>ROSE2</td>
<td><a href="http://idss.cs.put.poznan.pl/site/rose.html">http://idss.cs.put.poznan.pl/site/rose.html</a></td>
</tr>
</tbody>
</table>
### References

<table>
<thead>
<tr>
<th>Tool</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>4cMka2</td>
<td><a href="http://idss.cs.put.poznan.pl/site/4emka.html">http://idss.cs.put.poznan.pl/site/4emka.html</a></td>
</tr>
<tr>
<td>JSMAA</td>
<td><a href="http://www.smaa.fi/jsmaa/">http://www.smaa.fi/jsmaa/</a></td>
</tr>
<tr>
<td>jRank</td>
<td><a href="http://idss.cs.put.poznan.pl/site/146.html">http://idss.cs.put.poznan.pl/site/146.html</a></td>
</tr>
<tr>
<td>jMAF</td>
<td><a href="http://idss.cs.put.poznan.pl/site/139.html">http://idss.cs.put.poznan.pl/site/139.html</a></td>
</tr>
<tr>
<td>FINEVA</td>
<td>C. Zopounidis, N.F. Matsatsinis, and M. Doumpos. Developing a multicriteria knowledge-based decision support system for the assessment of corporate performance and viability: The FINEVA system</td>
</tr>
<tr>
<td>PREFDIS</td>
<td>C. Zopounidis and M. Doumpos. PREFDIS: A multicriteria decision support system for sorting decision problems</td>
</tr>
<tr>
<td>INVESTOR</td>
<td>C. Zopounidis and M. Doumpos. INVESTOR: A decision support system based on multiple criteria for portfolio selection and composition.</td>
</tr>
</tbody>
</table>

### 7.6.2 Ranking Methods and Tools

<table>
<thead>
<tr>
<th>MCDA Methods and Extensions</th>
<th>Links and References</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELECTRE II</td>
<td><a href="http://l1.lamsade.dauphine.fr/deal03/ens/bouyssou/Outranking_Mousseau.pdf">http://l1.lamsade.dauphine.fr/deal03/ens/bouyssou/Outranking_Mousseau.pdf</a></td>
</tr>
<tr>
<td>ELECTRE III</td>
<td><a href="http://l1.lamsade.dauphine.fr/deal03/ens/bouyssou/Outranking_Mousseau.pdf">http://l1.lamsade.dauphine.fr/deal03/ens/bouyssou/Outranking_Mousseau.pdf</a></td>
</tr>
</tbody>
</table>
# Appendix: MCDA - Methods and Tools References

<table>
<thead>
<tr>
<th>Method</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>REGIME</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>ORESTE</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>EVAMIX</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>TACTIC</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>MELCHIOR</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>MAPPAC</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>PRAGMA</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>PACMAN</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
</tbody>
</table>
### MARTEL AND ZARAS


### GRIP


### MACBETH


### CMBA


### ANP


### AHP


<table>
<thead>
<tr>
<th>Method</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>REASONING</td>
<td></td>
</tr>
</tbody>
</table>
| NAIADE          | J. Figueira, S. Greco, & M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer  
### 7.6 Appendix: MCDA - Methods and Tools References

<table>
<thead>
<tr>
<th>Method</th>
<th>References</th>
</tr>
</thead>
</table>
| **MAUT** | [http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt](http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt)  
| **MAVT** | [http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt](http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt)  
| **SMART** | [http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt](http://cbafiles.unl.edu/public/cbainternal/facStaffUploads/mcdm.ppt)  
[http://www.systemicplanning.dk/SMART.pdf](http://www.systemicplanning.dk/SMART.pdf) |
| **SMARTER** | [http://www.systemicplanning.dk/SMART.pdf](http://www.systemicplanning.dk/SMART.pdf)  
| **GREY RELATIONAL ANALYSIS** | Y. Kuo, T. Yang, G-W Huang. The use of grey relational analysis in solving multiple attribute decision-making problems. Computers and Industrial Engineering, Volume 55, Issue 1, August 2008, Pages 80-93  
<table>
<thead>
<tr>
<th>Reference</th>
<th>URL/Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTA I</td>
<td><a href="https://www.researchgate.net/publication/226057347_UTA_methods">https://www.researchgate.net/publication/226057347_UTA_methods</a></td>
</tr>
<tr>
<td>STOCHASTIC UTA</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>NATA</td>
<td><a href="https://en.wikipedia.org/wiki/New_Approach_to_Appraisal">https://en.wikipedia.org/wiki/New_Approach_to_Appraisal</a></td>
</tr>
<tr>
<td>NSFDSS</td>
<td><a href="http://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9364%282002%29128%3A3%28220%294">http://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9364%282002%29128%3A3%28220%294</a></td>
</tr>
<tr>
<td>PAPRIKA</td>
<td><a href="http://www.cs.put.poznan.pl/ewgmcd/a/newsletter/Fall11.pdf">www.cs.put.poznan.pl/ewgmcd/a/newsletter/Fall11.pdf</a></td>
</tr>
<tr>
<td>Fuzzy VIKOR</td>
<td>S. Opricovic. Fuzzy VIKOR with an application to water resources planning, Expert Systems with Applications, Volume 38, Issue 10, 15 September 2011, Pages 12983-12990</td>
</tr>
<tr>
<td>Method</td>
<td>Reference</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Reference</td>
<td>URL</td>
</tr>
<tr>
<td>-----------</td>
<td>-----</td>
</tr>
<tr>
<td>SMAA-P</td>
<td>R. Lahdelma, P. Salminen. Prospect theory and stochastic multicriteria acceptability analysis (SMAA), Omega, Volume 37, Issue 5, October 2009, Pages 961-971</td>
</tr>
<tr>
<td>Method</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>REMBRANDT</td>
<td><a href="http://orbit.dtu.dk/fedora/objects/orbit:40639/datastreams/file_6314365/content">http://orbit.dtu.dk/fedora/objects/orbit:40639/datastreams/file_6314365/content</a></td>
</tr>
<tr>
<td>RUBIS BEST CHOICE</td>
<td>Bouyssou, D., Marchant, T., Pirlot, M., Tsoukias, A., Vincke, P. Evaluation and Decision Models with Multiple Criteria. Springer US. 2006</td>
</tr>
<tr>
<td>TRICHOTOMIC SEGMENTATION</td>
<td><a href="http://cumincad.scix.net/cgi-bin/works/Show?ddss9215">http://cumincad.scix.net/cgi-bin/works/Show?ddss9215</a> B. Roy. A multicriteria analysis for trichotomic segmentation problems</td>
</tr>
<tr>
<td>UTA II</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
<tr>
<td>SMAA-3</td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
</tbody>
</table>
### References

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SMAA-O</td>
<td>R. Lahdelma, K. Miettinen, P. Salminen. Ordinal criteria in stochastic multicriteria acceptability analysis (SMAA), European Journal of Operational Research, Volume 147, Issue 1</td>
</tr>
<tr>
<td>MCDA Tools</td>
<td>Links and References</td>
</tr>
<tr>
<td>DECISSION</td>
<td><a href="http://decision-lab-2000.software.informer.com/">http://decision-lab-2000.software.informer.com/</a></td>
</tr>
<tr>
<td>LAB 2000</td>
<td></td>
</tr>
<tr>
<td>ELECCALC</td>
<td>L.N. Kiss, J.M. Martel, and R. Nadeau. ELECCALC An interactive software for modelling the decision maker’s preference</td>
</tr>
<tr>
<td>EQUITY</td>
<td><a href="http://www.catalyze.co.uk/index.php/software/equity3/">http://www.catalyze.co.uk/index.php/software/equity3/</a></td>
</tr>
<tr>
<td>EQUITY 3.2</td>
<td></td>
</tr>
<tr>
<td>EXPERT</td>
<td><a href="http://expertchoice.com/">http://expertchoice.com/</a></td>
</tr>
<tr>
<td>CHOICE</td>
<td></td>
</tr>
<tr>
<td>HIVIEW</td>
<td><a href="http://www.catalyze.co.uk/index.php/software/hiview3/">http://www.catalyze.co.uk/index.php/software/hiview3/</a></td>
</tr>
<tr>
<td>HIVIEW3</td>
<td></td>
</tr>
<tr>
<td>LOGICAL</td>
<td><a href="http://www.logicaldecisions.com/">http://www.logicaldecisions.com/</a></td>
</tr>
<tr>
<td>DECISION</td>
<td></td>
</tr>
<tr>
<td>SUPPORT</td>
<td></td>
</tr>
<tr>
<td>SYSTEM</td>
<td></td>
</tr>
</tbody>
</table>
### 7.6 Appendix: MCDA - Methods and Tools References

<table>
<thead>
<tr>
<th>Method/Tool</th>
<th>Website or Reference</th>
</tr>
</thead>
</table>
[http://www.m-macbeth.com](http://www.m-macbeth.com) |
| MIIDAS | J. Figueira, S. Greco, & M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer |
| MINORA | J. Figueira, S. Greco, & M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer |
| NAIADE | J. Figueira, S. Greco, & M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer |
| MULTICRIT (MCDMGDSS) | [http://nagesh-kumar.tripod.com/pubs/raju_iemkgp_98.pdf](http://nagesh-kumar.tripod.com/pubs/raju_iemkgp_98.pdf)  
<p>| PREFCALC | <a href="https://garage.maemo.org/projects/prefcalc/">https://garage.maemo.org/projects/prefcalc/</a> |
| RGDB | <a href="http://www.rgdb.org/idm/">http://www.rgdb.org/idm/</a> |</p>
<table>
<thead>
<tr>
<th>Reference</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTA PLUS</td>
<td><a href="http://www.lamsade.dauphine.fr/~mayag/links.html">http://www.lamsade.dauphine.fr/~mayag/links.html</a></td>
</tr>
<tr>
<td>VISA</td>
<td><a href="http://www.visadecisions.com">http://www.visadecisions.com</a></td>
</tr>
<tr>
<td>WEBHIPRE, HIPRE3+</td>
<td><a href="http://hipre.aalto.fi/">http://hipre.aalto.fi/</a></td>
</tr>
<tr>
<td>WINPRE</td>
<td><a href="http://www.sal.hut.fi/Downloadables/winpre.html">http://www.sal.hut.fi/Downloadables/winpre.html</a></td>
</tr>
<tr>
<td>MAKE IT RATIONAL</td>
<td><a href="http://www.makeitrational.com/">www.makeitrational.com/</a></td>
</tr>
<tr>
<td>TRANSPARENT CHOICE</td>
<td><a href="http://www.transparentchoice.com">www.transparentchoice.com</a></td>
</tr>
<tr>
<td>PRIORITY ESTIMATION TOOL</td>
<td><a href="http://priority.sourceforge.net">http://priority.sourceforge.net</a></td>
</tr>
<tr>
<td>SUPER DECISION</td>
<td><a href="http://www.superdecisions.com">http://www.superdecisions.com</a></td>
</tr>
<tr>
<td>WEB SUPER DECISION</td>
<td><a href="http://www.superdecisions.com">http://www.superdecisions.com</a></td>
</tr>
<tr>
<td>DECISION LENS</td>
<td><a href="http://decisionlens.com/">http://decisionlens.com/</a></td>
</tr>
<tr>
<td>ANP SOLVER</td>
<td><a href="http://kkir.simor.ntua.gr/anpsolver.html">http://kkir.simor.ntua.gr/anpsolver.html</a></td>
</tr>
<tr>
<td>INTELLIGENT DECISION SYSTEM</td>
<td><a href="http://e-ids.co.uk/">http://e-ids.co.uk/</a></td>
</tr>
<tr>
<td>1000MINDS</td>
<td><a href="http://www.1000minds.com">http://www.1000minds.com</a></td>
</tr>
<tr>
<td>VISUAL PROMETHEE</td>
<td><a href="http://www.promethee-gaia.net/software.html">http://www.promethee-gaia.net/software.html</a></td>
</tr>
<tr>
<td>PROMCALC</td>
<td><a href="http://www.promethee-gaia.net/software.html">http://www.promethee-gaia.net/software.html</a></td>
</tr>
<tr>
<td>TRIPTYCH</td>
<td><a href="http://www.stat-design.com/Software/Triptych.html">www.stat-design.com/Software/Triptych.html</a></td>
</tr>
<tr>
<td>DECIDEIT</td>
<td><a href="http://www.preference.nu/site_en/decideit.php">http://www.preference.nu/site_en/decideit.php</a></td>
</tr>
</tbody>
</table>
### 7.6.3 Choice Methods and Tools

<table>
<thead>
<tr>
<th>MCDA Methods and Extensions</th>
<th>Links and References</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ELECTRE I</strong></td>
<td><a href="http://l1.lamsade.dauphine.fr/dea103/ens/bouyssou/Outranking_Mousseau.pdf">http://l1.lamsade.dauphine.fr/dea103/ens/bouyssou/Outranking_Mousseau.pdf</a></td>
</tr>
<tr>
<td><strong>ELECTRE IS</strong></td>
<td><a href="http://l1.lamsade.dauphine.fr/dea103/ens/bouyssou/Outranking_Mousseau.pdf">http://l1.lamsade.dauphine.fr/dea103/ens/bouyssou/Outranking_Mousseau.pdf</a></td>
</tr>
<tr>
<td><strong>PROMETHEE V</strong></td>
<td>J. Figueira, S. Greco, &amp; M. Ehrgott, M. (Eds.). (2005). Multiple criteria decision analysis: state of the art surveys (Vol. 78) Springer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MCDA Tools</th>
<th>Links and References</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ELECTRE IS</strong></td>
<td><a href="http://www.lamsade.dauphine.fr/~mayag/links.html">http://www.lamsade.dauphine.fr/~mayag/links.html</a></td>
</tr>
<tr>
<td><strong>SANNA</strong> (for ELECTRE I)</td>
<td><a href="http://nb.vse.cz/~jablon/sanna.htm">http://nb.vse.cz/~jablon/sanna.htm</a></td>
</tr>
<tr>
<td>Visual Promethee (for PROMETHEE V)</td>
<td><a href="http://www.promethee-gaia.net/software.html">http://www.promethee-gaia.net/software.html</a></td>
</tr>
</tbody>
</table>
Chapter 8

Paper IV:
Extra-Functional Properties
Composability for Embedded Systems Partitioning

G. Sapienza, Séverine Sentilles, Ivica Crnkovic and Tiberiu Seceleanu.

Abstract

Modern embedded systems utilize the advances in heterogeneous platforms that enable implementing functions in software (SW) and hardware (HW) components. A proper configuration of SW and HW components can significantly improve the values of the extra-functional properties such as performance and energy savings. However, due to increasing application complexity, it is difficult to find the best combination of HW and SW components. The problem basically boils down to calculate, for a given architecture, the system properties from the components’ ones. In this paper, we address the problem of composable of EFPs at system level. Although in general this is not a solvable problem, we present that, under strictly specified constraints, it is possible to compose the system EFPs starting from the component ones. We start by detailing constraints related to the system architecture, platform and process development and, based on these constraints, we provide composition rules for different types of EFPs. We demonstrate the results through an industrial example.
8.1 Introduction

In recent years a visible trend appeared in embedded system domain: applications are implemented as software (SW) and hardware (HW) components and deployed on heterogeneous execution platforms with different executable units. The concurrent development process of HW and SW components is often called HW/SW co-design. The SW components, typically implemented in C/C++ are deployed on conventional CPUs, and HW components are implemented in a language that programs HW execution units. For example, VHDL (VHSIC Hardware Description Language) is used to program the components that are synthesised and deployed on a FPGA (Field-Programmable Gate Array) execution platform. Using these technologies, most of the functionalities can be implemented either as HW or SW components, but their non-functional properties (a.k.a. Extra-Functional Properties, EFPs) can be significantly different. For example, a parallel execution is superior in HW implementations, while system flexibility is easier to achieve with SW implementations.

Due to the increased application complexity, the number of SW and HW components becomes so large that it generates a new challenge: Which configuration of HW and SW components is optimal? The optimality refers mostly to non-functional requirements, and consequently to system EFPs executed by the components implemented either in SW or HW. A traditional way of analysing EFPs is to use a top-down approach in which EFPs are analysed after system implementation. In the case of HW/SW co-design, this approach becomes infeasible as the number of components grows significantly and the combination of their HW/SW implementations grows exponentially: for $n$ components with two possible implementations, there are $2^n$ possible configurations. Rather, an idea of a combination of top-down and bottom-up approach as used in Component-Based Software Engineering (CBSE) can be instead applied.

In this context, we start from existing components, by selecting particular implementation variants (HW or SW) and combine them in a HW-SW system. An effective design requires the possibility to model and predict system EFPs from the EFPs of the included components. This boils down to a composability question:

*Is it possible to derive system EFPs from components' EFPs?*

In general, the answer is no: Many EFPs are not even composable [1], and many of them are not analytically-composable (i.e. it is not possible to express the composition in a formal way). However, since embedded systems usually

---

1By “HW” components, we refer to implementations that are realised directly on an HW execution platform, e.g. an FPGA device.
have restricted architectures and many constraints related to their execution, by establishing these constraints on the system architecture (e.g. by only allowing a specific type of communication, using off-line scheduling algorithms, enforcing static memory allocation, etc.), many of the non-composable or non-analytically-composable EFPs become composable. Thus, it may be possible to derive their values at system-level from the component’s EFP values.

To answer the question stated above we:

1. identify assumptions that are valid for a large class of embedded systems (e.g. safety or mission-critical systems with restrictive resources such as energy, memory or CPU, real-time systems, and typically control systems), and

2. analyse EFPs and their compositions, and define the composition rules considering the given assumptions.

We analyse certain system EFPs that can be calculated from the EFPs of involved HW and SW components. In particular, we discuss differences in the composition of SW and HW variant combinations. The calculated EFPs can be used as basis for finding the best solution for a HW/SW configuration. In our previous work [2], we have classified EFPs based on existing standards (e.g. ISO/IEC9126) and quality models (e.g. McCall and Boehm) and analysed their impact on SW/HW co-design. Here we discuss a few of them, typically considered in embedded control systems design. It is worth noting that the EFPs considered in this work are not only runtime EFPs (i.e. those exhibited during the system execution), but also the ones related to lifecycle, and to business concerns and project constraints.

The paper is organised as follows. In Section 8.2, we formally define the component-based systems, and provide the assumptions that are valid for a specific class of embedded systems. In Section 8.3, we identify different types of EFP composability and their composition rules. Section 8.4 provides composition rules for certain amount of EFPs. Section 8.5 demonstrates the system EFPs in an industrial case study, namely a Wind Turbine application. Section 8.7 concludes the paper.

8.2 Preliminaries

8.2.1 System Model and EFPs

In our research, we consider heterogeneous embedded systems that can be modelled as a set of interconnected components, which are to be deployed on a
known platform. Using the definition from [1], we define such a system $S$ as:

$$ S = \langle P, C, B \rangle $$

(8.1)

where $P$ is the system platform, $C$ is a set of components $C_i$, and $B$ is the set of bindings $B_i$ between the components. Together, $C$ and $B$ describe the component-based architectural description of the system.

The system platform $P$ is the physical hardware which the system will be deployed on. The platform provides also some of the constraints of the project which do not change once decided. A component $C \in C$ is specified by

$$ C = \langle I, P \rangle $$

(8.2)

where $I$ is its Interface, and $P$ the set of properties. For each component $C$, that is represented as a model (i.e. a specification), there can be more implementations, i.e. component variants, implemented as SW or HW.

$$ C = \{ C_i \} : i = 1..n $$

(8.3)

$$ C_i = \langle I, P_i \rangle , P_i = \{ p_k, k = 1..m \} , P_i \subseteq P $$

(8.4)

$$ p_k = \{ \text{type}_k, \text{value}_k, \text{context}_k \} $$

(8.5)

Above, $C_i$ is the $i$-th implementation variant associated to the component $C$ and $n$ is the total number of variants for that component. Each variant implements the same interface $I$, but $P_i$, the set of properties that contains the EFPs $p_k \in P_i$, can vary from variant to variant. A definition of $p_k$ is elaborated in details in [3], and we adopt this definition to: $p_k$ is defined by its type (e.g. “response time”), value (e.g. “3 msec”), and context (e.g. execution unit, operating system (OS) ). Note that this definition allows different properties and different property values for different component implementations. In particular, HW variants are likely to have certain properties that are specific only for HW, and SW variants some properties that are specific only for SW.

Figure 8.1 illustrates the concepts of components and bindings with an abstract representation of the component-based architectural description of a system (left-hand side). For each component $C_i$ in the left-hand side of the figure, the set of its available variants $C_{ij}$ are presented in the table (in the center). The variants have their EFPs calculated for the targeted platform. The deployed architecture on the platform with its communication channels is visible in the
deployment model (right-hand side). $C_{11}$ and $C_{33}$ are HW variants while $C_{21}$ and $C_{n2}$ are SW variants.

![Figure 8.1: A component-based architectural description of a system with components (left-hand side) and their variants and related extra-functional properties (center) and an example of deployment model (right-side side).](image)

### 8.2.2 Assumptions

The technologies used for HW/SW co-design and implementation, build the systems that have many restrictions due to several reasons. One is the need for resource-efficient systems (e.g. systems with low energy consumption as they rely on battery power, or systems with restricted execution capabilities due to high production volume with low production costs), simple operating systems and execution semantics (which is typically for safety-critical systems). We consider the restrictions that are related to a) the HW platform on the OS used, b) execution semantics in technologies and component models typically applied in HW-SW applications, and c) the project development process. Project-related assumptions are defined in a restricted manner to simplify the identification of project-related requirements, but these restrictions do not influence the generality of the reasoning about project-related EFPs.

For the purpose of this study, we set the following assumptions.

#### Platform-related

The platform that we consider is a system on chip (SoC) solution supported by both a configurable logic part FPGA and CPU processing units which share a fixed communication channel. Examples of such platforms include the Xilinx’s Zynq-7000 family\(^2\) or the Altera’s FPGA-SoC systems\(^3\). We do not consider

---

\(^3\)https://www.altera.com/products/fpga/arria-series.html
8.2 Preliminaries

distributed systems, i.e. HW and SW components are deployed on the same platform and share the same resources. A single-user real-time operating system is used. The OS uses a simple scheduling algorithm (time-triggered), the static memory is allocated at system start, and there is no dynamic deployment of components.

**Interface and communication**

the functional interface of the components is specified through a set of input and output data ports. It is shared by all the variants associated to a given component. Components communicate between each other in a unidirectional pipe & filter style, which allows analysing the flow of data. Pipe & filter is the predominant interaction style used by component models for embedded and real-time systems [1].

Many of component models separate the data from the control flow which facilitates static analysis of EFPs related to timing properties. In this work, we further restrict the binding possibility by assuming that the control flow strictly follows the flow of data.

**Execution semantics**

The components follow a Read-Execute-Write (REW) semantics, characterized as follows:

- Each component is initially passive unless triggered by receiving data. A component with multiple input ports will start its execution upon the arrival of data on all ports.

- When triggered, the component executes the “read phase”, in which all the data available at the input ports are transferred to local variables within the component.

- Then, the component enters the “execution phase”, in which it uses the read data to perform its functionality.

- During the subsequent “write phase”, the component provides simultaneously all of the data on the output ports and returns to the passive mode.

Adopting such semantics increases the predictability and facilitates the analysis of EFPs by allowing to analytically model component executions with input
and output functions [4]. Component models such as ProCom [5], Rubus\(^4\) and modelling languages such as EAST-ADL\(^5\) assume this semantics.

The required resources (e.g. cache and DRAM memory, buses, etc.) for a component are considered to be available at each component execution. Similarly, the required data are expected to be available in a similar way for each execution (e.g. data are always available in the memory cache).

**Project-related**

Project constraints and business concerns are strictly dependent on development management factors such as legacy, standards and legislations, business priorities, cost constraints, platforms constraints, etc. Most of the EFPs belonging to this class (a classification is given in [2]) are project-specific, hence they might vary from project to project. In order to limit the scope of our research, we consider that costs and times related to different development phases are not affected by non-linearities related to the number of components or their integration.

### 8.3 Analytical calculation of system EFPs

This section presents our proposed approach to analytically derive system properties starting from the component properties acting in isolation and using knowledge related to the system architecture and underlying platform. Three aspects are considered: (i) composability of the EFPs, (ii) composition patterns, and (iii) binding types.

#### 8.3.1 EFP composability

The composability of a property is defined as its ability to being composed by other properties. In a component-based approach, composability thus refers to a property composed of the properties of the involved (i.e. interacting) components.

\[
p(C_1, C_2, \ldots, C_n) = p(C_1) \oplus p(C_2) \oplus \ldots \oplus p(C_n)
\]  


By considering [1, 6] as starting points, we propose here a system EFP classification with respect to composability. We will then classify the properties and identify their composition rules in the next section.

We start by reasoning that system EFPs can be considered as:

**Emergent or non-emergent:** Emergent properties are properties which are only visible at system level and non-existing at component level. This is similar to physical properties available in real life: saltiness is a property of something having salt, but neither the ions of sodium nor chloride that are the based components to build up salt are salty. There are many properties of software and software intensive systems of this type (e.g. safety, resilience). A formal definition of emergent properties can be given as:

\[
P_S - a set of system properties; \\
P_C - a set of component properties; \\
(p \in P_S) \Rightarrow (p \notin P_C) - emergent property p
\]

**Non-emergent** properties are those which are visible at both system and component levels, i.e. both the system and the components have these properties.

\[
(p \in P_S) \Rightarrow (p \in P_C) - non-emergent property p
\]

**Composable and non-composable:** Composable properties are those which values can be derived from other properties. Composable system properties are compositions of the properties of the components. An example of such property is response time. **Non-composable** properties are those which cannot be derived from other EFPs. These system properties are always emergent. Two examples of non-composable properties are safety and security.

Focusing now on composable system EFPs only, we divide them in the following categories:

**Directly composable properties:** are those which can be directly derived from the same property of the involved components. In many cases, they can be analytically expressed as a function. In practice, only a few system EFPs are directly-composable, and they are strictly dependent on specific architectures, used technologies or other constraints. Examples of such properties are memory footprint, execution time in a pipe and filter style with components in a single thread. Formally:
\[ p_i \in P_S \land p_i \in P_C \]
\[ p_i(C_1, C_2, ..., C_n) = p_i(C_1) \oplus p_i(C_2) ... \oplus p_i(C_n) \]

(8.7)

Indirectly composable properties: those properties are the result of the composition of the same components’ properties and of other factors. Examples of these properties include system design time and testing time, which are not only function of the components’ design and test time properties but also depend on the system complexity. In the formal definition, we introduce different assets \( A_i \) that are a part of the composition rule.

\[ p_i \in P_S \land p_i \in P_C \]
\[ p_i(C_1, C_2, ..., C_n) = p_i(C_1) \oplus p_i(C_2) ... \oplus p_i(C_n) \oplus (A_1, A_2, ...) \]

(8.8)

Derived properties: those properties can be composed by different properties of the involved components and possibly other factors, for example the architecture. These system properties are emergent properties. An example of such property is the return on investment (ROI) which results of different measures and properties (e.g. efforts, costs, price, outcome, etc.). We formally defined them as:

\[ p_i \in P_S; p_k, p_j \in P_C \]
\[ p_i(C_1, C_2, ...) = (p_k(C_1), p_j(C_1)) \oplus (p_k(C_2), p_j(C_2)) ... \oplus (A_1, A_2, ...) \]

(8.9)

In many cases, refining the composition rules or relaxing the assumptions can change the composition category. For example, a directly composable property can become a derived property. e.g. the response time of a service can be calculated from the execution time of the involved components in a simple execution semantic, while in case of using a specific scheduling policy, the scheduling algorithm will be included in the calculation.

8.3.2 Composition patterns

Thanks to the pipe & filter interaction style and the REW semantics, certain composition patterns can be identified from the connection types between com-
ponents, i.e. from the system architecture. Inspired by the equivalent electrical
circuit combinations for resistors, we define here three composition patterns
that can be applied to facilitate the calculation of property values. By analysing
how components are connected in pairs, we decompose the system architecture
in a succession of basic patterns.

- the sequence pattern: the components are linearly connected, and execute in
  sequence. In Figure 8.2 a), the output from one component ($C_1$) is directly
  used as input by the next component in the sequence ($C_2$);

- the parallel pattern: the components execute in parallel. This can be true
  parallelism when $C_1$ and $C_2$ are two HW variants, or when $C_1$ is a HW
  variant and $C_2$ is a software variant (or vice-versa). Alternatively, this pattern
  can represent pseudo-parallelism when $C_1$ and $C_2$ are two SW variants. This
  difference between true and pseudo parallelism is generally reflected in the
  composition rules. Figure 8.2 b) depicts this case.

- the loop pattern: the output of a component is fed back to the preceding
  component (Figure 8.2 c). Components connected in a loop store output
  data in a buffer. The buffer can be the memory for SW variants or physical
  buffering in the FPGA for HW variants.

![Figure 8.2: Base composition patterns](image)

8.3.3 Binding types

In order to support both HW/SW technologies, the considered platform must
provide means to transfer the output from one technology domain to the other,
which impacts the resulting values for some of the EFPs at system level. Typ-
ically, it is important to know the binding types to derive time-related proper-
ties at system level. For instance, communication across the link between the
CPU and the FPGA or via the memory provide different communication times. Therefore, the specific connections between variants must be identified and used in the composition rules.

Based on the taken assumptions, we can define four types of bindings between the variants, as follows:

- **SW-SW**: both variants are implemented in SW. Communication is performed via shared memory.
- **SW-HW**: a SW component is connected to a HW component, with data sourcing in the SW component. Communication is performed via the CPU-FPGA interface.
- **HW-SW**: a HW component is connected to a SW component, with data sourcing in the HW component. Communication is performed via the FPGA-CPU interface.
- **HW-HW**: both variants are implemented in HW. Communication is performed via specific connections realised as FPGA units.

The binding choices and the pipe & filter interaction style ensure the atomic data transfer between components of the same variant type, while communication between HW and SW components require a certain amount of time.

### 8.4 System Composition rules

Based on the classification, assumptions and the component-based model presented in Section 8.3, we identify a number of composition rules to derive the system EFPs from the EFPs of the components. Specifically, we focus here on providing rules to calculate the system EFPs using the value(s) of the $p_k$ of each component $C$, defined in (8.3) and (5). We will calculate the values of system $p_k$ for different EFPs. We start by presenting some examples of system EFPs related to the indirectly-composable class, we continue with the derived system EFPs and we conclude the section with directly-composable EFPs (where we refer to some of the indirectly-composable properties, too).

We should mention that we focus on a subset of composable EFPs that are important for HW/SW co-design and for which it is relatively easy to provide the composition rules with the given constraints. The goal is to eventually provide our partitioning mechanism [7] with values for decision making. There is a large body of knowledge on EFPs that is not addressed here.
8.4 System Composition rules

8.4.1 Indirectly-composable system properties

We describe below a set of EFPs that represents design and test efforts (i.e. lifecycle and project constraints/business concerns-related properties), and response time (related to run-time properties).

**System Design Cost (SDC).** In [2], the component design cost \((CDC)\) is defined as the one-time monetary costs needed to design a component. At system level, it can be calculated as a function of the design cost of each single component involved and of additional design costs. Examples of such additional costs are the costs of the architectural design, platform related activities, or design costs required to realize the communication between the HW and SW variants. Such types of costs are specific to a given platform and system architecture. The \(SDC\) can be calculated as follows:

\[
SDC = \sum_{i=1}^{n} CDC_{Ci} + EDC,
\]

where the \(CDC_{Ci}\) is the component design time for the \(i\)-th component; \(EDC\) is the Extra Design Cost related to the design of the system architectural design and the platform related activities; \(n\) is the total number of components. In the simplest case, the EDC can be considered as a constant, but it might take into account non-linear aspects in a more refined analysis. However, these latter analyses are not addressed in this work. Similar considerations apply in the other formulas below.

**System Testing Lead Time (STLT).** The testing lead time is the (calendar) time needed to perform the test of a component. It is calculated as a function of the testing lead time of the components \((CTT)\) involved and the additional testing time required to carry out the co-verification of the implemented HW/SW variants for a given system architecture. Examples of additional testing time include the time for preparing the test-bench, the test scripts, the generations of the test scripts, etc. The \(STLT\) is then derived as follows:

\[
STLT = \sum_{i=1}^{n} CTT_{Ci} + ETT,
\]

where the \(CTT_{Ci}\) is the component testing time for the \(i\)-th component, \(ETT\) is the extra testing time related to the co-verification aspects and \(n\) is the total number of components involved.

**System Development Environment Cost (SDEC).** The system development environment cost is defined as the overall monetary cost related to the development items. Development items include the tool licenses, prototyping and eval-
ulation boards, a hardware-in-the-loop (HIL) testing system, among many other elements that are needed for the entire development of the system. Each component might have associated one or more so called environment items. By associating to each environment item a predefined cost, the $SDEC$ can be calculated as follows:

$$SDEC = \sum_{E=1}^{k} \sum_{i=1}^{n} \left( \frac{P_{Ci,E}}{n_E} \times \text{cost}_E \right),$$  (8.12)

where $E$ is the environment item; the $P_{Ci,E}$ is a boolean EFP related to the environment associated $i$-th component. It indicates if the $C_i$ component uses the environment item $E$; $n_E$ is the number of components using the specific environment item $E$. The cost of a specific environment item is indicated as $\text{cost}_E$.

**System Upgradability ($SUpp$).** In [2], the upgradability is defined as the extent to which a component is capable of being improved in functionality by the addition or replacement of a component part. Thus, the system upgradability can be calculated as the sum of the number of the component parts that can be improved divided by the total number of components. This gives the average upgradability of the systems (expressed in percentage). However, at system level, we also need to take into account the impact of the upgrade of a given component with respect to the other components in the system. This is, for instance, due to the communication or memory architecture of the system. This would be the case when a component upgrade requires additional modifications on how the data are buffered in the programmable-logic fabric. We calculate the system upgradability as follows:

$$SUpp = \frac{1}{n} \sum_{i=1}^{n} \min(\text{UpC}_i, \text{UpI}_i),$$  (8.13)

where the $\text{UpC}_i$ is the component upgradability for the $i$-th component (expressed in percentage); $\text{UpI}_i$ is the upgradability impact of the $i$-th component with respect to the system (expressed in percentage as well); $n$ is the total number of components.

**System Response Time (SRT).** SRT is defined as the time needed for a signal coming from the input to produce a change at the system output. We compute the $SRT$ from the execution time ($\text{ExT}$) of each processing component and the communication time between the components. The $SRT$ computation time

\[\text{The processing components may vary in different usage scenarios, or depending on input data.}\]
depends on: a) how the components are executed (sequentially or in parallel) and b) the binding type of the given components.

Table 8.1: System Response Time Formulas

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Binding Type</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequential</td>
<td>SW-SW</td>
<td>(RT_{SW-SW} = ExT_{C_a} + ExT_{C_b} + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Sequential</td>
<td>SW-HW</td>
<td>(RT_{SW-HW} = ExT_{C_a} + ExT_{C_b} + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Sequential</td>
<td>HW-SW</td>
<td>(RT_{HW-SW} = ExT_{C_a} + ExT_{C_b} + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Sequential</td>
<td>HW-HW</td>
<td>(RT_{HW-HW} = ExT_{C_a} + ExT_{C_b} + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Parallel</td>
<td>SW-SW</td>
<td>(RT_{SW-SW} = max(ExT_{C_a}, ExT_{C_b}) + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Parallel</td>
<td>SW-HW</td>
<td>(RT_{SW-HW} = max(ExT_{C_a}, ExT_{C_b}) + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Parallel</td>
<td>HW-SW</td>
<td>(RT_{HW-SW} = max(ExT_{C_a}, ExT_{C_b}) + CT_{C_a,C_b})</td>
</tr>
<tr>
<td>Parallel</td>
<td>HW-HW</td>
<td>(RT_{HW-HW} = max(ExT_{C_a}, ExT_{C_b}) + CT_{C_a,C_b})</td>
</tr>
</tbody>
</table>

Figure 8.3 shows how the formulas of Table 8.1 are applied for calculating the \(SRT\) when variants are connected in sequence or parallel. Following the control flow, \(ExT_{C_a}\) is the execution time of the first component and \(ExT_{C_b}\) is the one related to the second component, while \(CT_{C_a,C_b}\) is the extra execution time related to the data communication.

Figure 8.3: Sequence and Parallel transformations. In a) and b) we show how to apply the formulas of Table 8.1.

A component-based model compliant to the metamodel proposed in [7] can be represented as a graph, where each component is a graph vertex and the
binding represents a graph edge. The total $SRT$ of the model is calculated by applying existing algorithms (e.g., the depth-first search algorithm [8] or the breadth-first search algorithm [9]) for traversing the graph.

System Power Consumption (SPC). The system power consumption can be calculated as the sum of the single component power consumption $CPC$, plus the additional power related to the system architecture and platform. An example of this additional power related to the platform is the static power consumption related to the FPGA or the extra power consumption related to the bus used for the communication between the HW and SW variants. The $CPC$ can be calculated as follows:

$$SPC = \sum_{i=1}^{n} CPC_i + EP,$$

(8.14)

where the $CPC_i$ is the component power consumption for the i-th component; $EP$ is the extra power consumption related to the system architecture and platform; $n$ is the total number of components. It has to be noted here that power consumption is a value dependent on multiple factors. Thus, the numbers used in such context are estimations which are very often highly inaccurate. However, as the errors are considered to be evenly distributed, this EFP calculation can still give good relative configuration solutions. Similar considerations and composition rules (see $SDC$, or $STLT$ EFPs) can be applied for calculating other EFPs as for instance the costs or lead times related to the system implementation cost or the system maintenance lead time.

8.4.2 Derived system properties

The following EFPs are derived from other properties.

Total Development Cost (TDC). Considering the different development phases (as for instance the requirement analysis phase, design phase, implementation phase, verification and validation phase), the system total development cost is a function of the component cost for each phase (e.g. the System Design Cost $SDC$), the $SDEC$ (which is calculated using the formula presented in Section 8.4.1) and the cost related to the overall management of the project. It can be calculated as:

$$TDC = \left( \sum_{P=1}^{k} \sum_{i=1}^{n_P} cost_{C_iP} \right) + SDEC + cost_{ST},$$

(8.15)

where $P$ indicates a development phase, and it goes from 1 to $k$ (number of total development phases); $n_P$ is the number of components involved in the
calculation for a given development phase; \( \text{cost}_{CiP} \) is the cost of the i-th component associated to a specific development phase; and \( \text{cost}_M \) is the overall cost related to the project management.

**Total Development Lead Time (TDLT).** Similar to the considerations done for the \( (TDC) \) calculation, for the different development phases the average system total development lead time \( TDLT \) can be expressed as a function of the component lead time in each phase (e.g. the \( \text{STLT} \)) and the lead time related to the overall management of the project, both terms are divided by the number of resources (e.g. engineers, project managers) involved in the development.

Consequently, the textit{TDLT} will be expressed as:

\[
TDLT = \left( \sum_{P=1}^{k} \sum_{i=1}^{n_P} \frac{LT_{CiP}}{R_P} \right) + \frac{LT_M}{R_M},
\]

where \( P \) indicates a development phase, and it goes from 1 to \( k \) (number of total development phases); \( n_P \) is the number of components involved in the calculation for a given development phase; \( LT_{CiP} \) is the lead time of the i-th component associated to a specific development phase; \( R_P \) is the number of e.g. engineers involved in each phase; \( LT_M \) is the lead time related to the project management; \( R_M \) is the number of the managers leading the project.

### 8.4.3 Directly-composable system properties

The system properties are calculated only from the the same properties of the involved components.

**System Traceability (STr).** In accordance with the McCall’s quality model [10], the traceability is defined as the ability to trace a component design representation or implementation back to requirements. Assuming the component traceability as a link between a component design representation and at least one requirement, the system traceability can be calculated as the logic “AND” of the single traceability of each component:

\[
STr = \prod_{i=1}^{n} CTr_{Ci},
\]

where the component traceability of \( (CTr_{Ci}) \) for the i-th component is expressed as a boolean and \( n \) is the total number of components.

**System Design Cost (SDC).** The following EFP has been discussed above, in the indirectly-composable class. However, under some assumptions the system design cost calculation can be simplified, and this property can be considered
a directly composable one. Under the assumption that the extra design related to the architecture and platform related aspects are negligible, the $SDC$ computation can be simplified:

$$SDC = \sum_{i=1}^{n} CDC_{C_i}, \quad (8.18)$$

where the $CDC_{C_i}$ is the component design time for the $i$-th component; $n$ is the total number of components. For instance, this might be the case where the drive routines needed are already available, and the tools used for the design are able to automatically configure the HW/SW communication interface and the handling of the memory. As a consequence, the design time required for these tasks is insignificant. Similar reasoning, which lead to a simplification of the related composition rule, can also be carried out for other system EFPs such as the system implementation cost or system testing cost.

**System Static Memory (SSM).** Typically, the static memory is allocated in a compilation phase, and can be calculated and specified for each component. Similarly to [6], we can calculate the system static memory ($SSM$) as the sum of the static memory related to the software variants plus the additional static memory usage due to for instance the parameterization of the system interface or the configuration of the real time operating system. The $SSM$ can be expressed as follows:

$$SSM = \sum_{i=1}^{n} CSM_{C_i} + EM, \quad (8.19)$$

where the $CSM_{C_i}$ is the component static memory for the $i$-th component; $EM$ is the extra static memory utilization; $n$ is the total number of components.

**System FPGA Area (SFA).** The system FPGA area utilization ($SFA$) is a function of the area utilization of the HW variants and the additional area utilization that can be required for instance for the communication. It is usually expressed in number of gates allocated and can be calculated as:

$$SFA = \sum_{i=1}^{n} CFA_{C_i} + EA, \quad (8.20)$$

where the $CFA_{C_i}$ is the FPGA area utilization for the $i$-th component; $EA$ is the extra FPGA area utilization; $n$ is the total number of components.

---

8.5 Industrial Example

We present an industrial example to demonstrate how the set of composition rules discussed in Section 8.4 can be applied. Our aim is to show the feasibility of composing the system EFPs starting from the EFPs of the involved components. For this purpose, we use an industrial prototype, a small wind turbine control system, developed within the scope of the iFEST EU project.

Wind energy sources are fast-growing and more sophisticated turbine systems have to be developed to meet customer demands. Consequently, in order to fulfill requirements such as new functionalities, performance, time-to-market, and maintainability, the development of such control systems become more challenging. The main task of a wind turbine system is to convert the rotational mechanical energy of the rotor blades (i.e., mechanical components of a wind turbine) caused by the wind into electrical energy, which will be later re-distributed via a power network.

Compliant with the metamodel proposed in [7], we modelled in Figure 8.4 the application (here referred as controller) as a number of interconnected components. The PI_Controller, Filtering, Main Controller and Pitch Estimator components are taken from a library of existing components. For each component, to select the most suitable HW and SW variants we applied the method proposed in [7], which from all possible $2^7$ deployment configurations gives the best HW and the best SW variant of each component. The associated component EFPs were calculated, estimated or simulated.

![Diagram of wind turbine controller component-based model](image-url)

Figure 8.4: High-level overview of the wind turbine controller component-based model (modelled using the MathWorks Simulink).

In order to show the applicability of the proposed composition rules, we here consider three possible system deployment configurations: 1) a HW sy-
stem realization (Only-HW_Conf) 2) a SW system realization (Only-SW_-Conf), and 3) a system realization which includes both HW and SW component implementations (HW-SW_Conf). In Figure 8.5, a simplified overview of the component’s EFPs is available on the left-hand side. For each component, the EFPs values of the HW and the SW variants are given. On the right-hand side, a table showing the calculated system EFPs is provided. The arrows are used to highlight the relations between the component’s EFPs and the analytically composed system EFPs. To develop the controller, we used the tools and platform listed in Table 9.6; the 5th column shows the Requirement Elicitation (RE), Design (D), Implementation(I), Testing(T) phases in which the item was used. No real-time operating system was utilized.

For each system EFP, we discuss below the relevant concepts, assumptions or constraints that affected the calculations of the composition rules:

– indirectly-composable EFPs

System Design Cost (SDC). Besides the cost of the single component (which for instance includes the cost for designing it from scratch, or the cost of intellectual property cores), we took into account the costs related to the design of the system communication via the Advanced eXtensible Interface (specifically the AXI 4-lite [11]) and the cost related to the design decisions upon the most suitable memory architecture (e.g. related to the need of instantiate an AXI-DMA controller or more AXI4 Stream FIFO cores).
8.5 Industrial Example

System Testing Lead Time (STLT). For this calculation, the estimated time needed the setting of the test-bench, the time needed to execute the test scripts and the time to analyze the test results. The test scripts were automatically generated using the MaTeLo tool [12], and no additional time was required.

System Development Environment Cost (SDEC). To calculate this EFP, we used the item identifier (ID) values and the development phase (Dev. Phase) shown in Table 9.6, where for the different phases we assigned the following values: RE = 1; D = 2; I = 3; and T = 4.

Table 8.2: Development Environment

<table>
<thead>
<tr>
<th>Item</th>
<th>Name</th>
<th>Provider</th>
<th>Type</th>
<th>Dev. Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application LifeCycle Management (ALM)</td>
<td>Hewlett-Packard (HP)</td>
<td>Tool</td>
<td>RE, T</td>
</tr>
<tr>
<td>2</td>
<td>Enterprise Architect</td>
<td>Sparx System</td>
<td>Tool</td>
<td>RE, D</td>
</tr>
<tr>
<td>3</td>
<td>Matlab</td>
<td>MathWorks</td>
<td>Tool</td>
<td>D, I, T</td>
</tr>
<tr>
<td>4</td>
<td>Simulink</td>
<td>MathWorks</td>
<td>Tool</td>
<td>D, I, T</td>
</tr>
<tr>
<td>5</td>
<td>Embedded Coder</td>
<td>MathWorks</td>
<td>Tool</td>
<td>I</td>
</tr>
<tr>
<td>6</td>
<td>HDL Coder</td>
<td>MathWorks</td>
<td>Tool</td>
<td>I</td>
</tr>
<tr>
<td>7</td>
<td>Vivado Design Suite</td>
<td>Xilinx</td>
<td>Tool</td>
<td>D, I, T</td>
</tr>
<tr>
<td>8</td>
<td>MaTeLo</td>
<td>All4TEC</td>
<td>Tool</td>
<td>T</td>
</tr>
<tr>
<td>9</td>
<td>Microsoft Project</td>
<td>Microsoft</td>
<td>Tool</td>
<td>RE, D, I, T</td>
</tr>
<tr>
<td>10</td>
<td>Zynq ZC702</td>
<td>Xilinx</td>
<td>Platform</td>
<td>D, I, T</td>
</tr>
</tbody>
</table>

These values will be the same for all of the other composition rules. The Microsoft Project and the Application LifeCycle Management tools were already available, since used in previous development projects. In Figure 8.5 only the items that differ from component to component are reported.

System Upgradability (SUP). Since the system has been recently developed, for this calculation, we estimated the extent to which each variant might be upgraded in the future, and the average impact of the upgrades on the communication architecture which is supposed to be upgradable at 80%. These values are respectively the % of CUp and the % of UpI. In Figure 8.5, only the environment items that differ from variant to variant are shown.

System Response Time (SRT). In order to calculate the SRT, the following information were taken into account: a) FPGA frequency is equal to 8 kHz; b) the CPU frequency is equal to 800 kHz. For variants executed on the same technology, CT is considered negligible. For $CT_{SW-HW}$ and $CT_{HW-SW}$, we considered the average amount of data that can be transferred per clock cycle (for a 32-bit word-length for the given platform, it was estimated to be 125 µs).
– derived EFPs

*Total Development Cost (TDC).* We started by summing up the costs of each component (the right-hand side of Figure 8.5) for the different phases, i.e. the RE, D, I and T. We then added the cost obtained by the calculation of the SDEC, and finally calculated the cost related to the management of the project.

*Total Development Lead Time (TDLT).* To calculate this property, we considered the number of human resources working on the project. The project team consists of 2 engineers for the RE phase, 2 designers for the D phase, for 2 developers for the I phase, 1 test engineer for the T phase. Only one manager led the project.

– directly-composable EFPs

*System Traceability (STr).* In this project we had the benefit of an in-house developed tool extension able to trace the design between HP ALM [13] and Simulink [14]. This increased the traceability of the process. However, as it can be noticed from the right-hand side of Figure 8.5, some components are not backward traceable. Consequently, it affected the overall traceability.

*System Static Memory (SSM).* For this calculation, we considered the memory related to the system parametrization. This is a software property. As a result, only the values for the SW variants are shown in Figure 8.5.

*System FPGA Area (SFA).* To calculate these values, we considered the information provided by Xilinx, where 1 Logic Cell is approximately equal to 15 ASIC Gates\(^9\). As a consequence, in Figure 8.5, the values of the HW variants are expressed in equivalent ASIC gates unit.

For the sake of completeness, we provide here a prioritization of the three deployment configurations based on the requirements analysed in [2]. We use a multi criteria decision analysis (MCDA), which fits well with solving decision problems dealing with a finite number of alternatives (i.e. the deployment configurations) and many conflicting EFPs. We use the System for ANalysis of Alternatives (SANNA) tool [15] and we prioritize the system EFPs according to the values provided in Figure 8.5 in the “Priority” column. The outcome is given in the “Configuration Ranking MCDA” column.

### 8.6 Related Work

Albeit having already been addressed in multiple studies in the past (for instance [16, 17]), composing system EFPs is still a key challenge when deve-
loping software component-based systems [6, 18]. Though the problem was intensively studied from the functional and non-functional perspective (for example [1, 6, 19]), to the best of our knowledge, no previous work was carried out taking a software and hardware combined perspective. In the software domain, several proposed solutions for EFP analysis only focus on one or very few EFPs at the time and require multiple detailed and low-level information making them loosely applicable in a large context or when requiring to take decisions by trading-off many, possibly, conflicting EFPs. For example, for timing analysis a compositional timing analysis for resource-sharing systems is proposed in [20], while [21] proposes a response time analysis for fixed priority preemptive systems. In difference to our approach, these works do not use the existing EFPs values of the components considered in isolation but they model the entire system for solving the specific problem. For HW/SW co-design, there are studies that address component compositions (e.g. [22]) via their modelling platform (such as Ptolemy II [23]), but the composition is based on the analysis of functional and timing properties only. In difference to these works, we propose here a general and high-level approach for EFP composability with strong assumptions on system constraints that are however usual in many types of embedded systems. This is applicable for HW and SW component realizations and can be applied to take development decisions upon many EFPs.

8.7 Conclusions

In this paper, we have analysed the possibility of computing system EFP values from the values of the components’ EFPs. One specificity of this work is that the components considered here are implemented either as SW components running on a CPU, or HW components implemented on FPGA fabric. Another specificity relates to the architectural characteristics of a class of embedded systems with a component model that uses pipe and filter interaction style, has a read-execute-write execution semantics, and uses specific interaction channels. These strong constraints make many system EFPs composable. In difference to our previous work [2] where we have addressed and classified the EFPs of interest for partitioning into three main categories i.e. lifecycle, runtime and project/business-related, we have here classified different types of EFPs with respect to their compositions: directly composable, indirectly composable, and emergent EFPs. For each of these types, we have presented calculations for some of EFPs that are concerns of importance in embedded systems design.
While some compositions are simple and straightforward, some of them require more in depth analysis, while still derivable. We have also presented an industrial use case a model of a wind turbine control system. We have demonstrated that: a) we can use a component model in which we combine HW and SW variants, and b) by placing strong enough constraints on the system, we can reason about and even calculate the system properties from the properties of the composing SW and HW variants.

Future work includes three main directions. First, a further refinement of EFPs and their compositions. In several cases we have filtered out some architectural details of the system. Although in embedded systems typically simple system architectures are used, there are architectural details that have impact on many EFPs which we have not addressed in this study. By refining the architecture and by relaxing some of the constraints, the nature of the compositions will be changed, and the relation between the constraints and the compositions can be further analyzed. The second direction of the future work considers finding the “best” HW and SW component configuration, i.e. a configuration that would best fit to the specified requirements. In our previous work [7], we developed a method for selecting the best HW and SW component variants with respect to the defined component EFPs. This work will be extended to the system EFPs by using the composition rules. The third direction is to apply and validate the proposed composition rules on the development of a more complex industrial application in the automation domain. Thus, the feasibility, and the practical aspects, such as the balance between efforts and the results of the analysis can be proved and respectively validated.

Acknowledgment

This research is supported by the Knowledge Foundation through ITS-EASY, Industrial Research School in Embedded Software and Systems, ORION project, and by the Swedish Foundation for Strategic Research through the RALF3 project.

References


Chapter 9

Paper V: Methodology for Embedded Systems Partitioning based on Multiple Criteria Decision Analysis

G. Sapienza, N. Meli and J. Eriksson

Abstract

Platforms with different computation resource, e.g. CPUs and FPGAs, known as heterogeneous platforms, are becoming one of the first choices to deploy performance-requiring embedded applications. On this technology, functionalities can be implemented either as hardware (HW) or software (SW) components. In this paper, we propose a new methodology, called MULTI PAR, which is able to provide optimal partitioning solution(s) with respect to many system properties based on Multiple Criteria Decision Analysis. We provide an overview of the methodology. MULTI PAR is based on a bottom-up principle, first, to find the most suitable SW and HW implementations based on their properties, and then applying compositional rules and other means uses the system properties to find the best partitioning solutions. In addition, we show the feasibility of the proposed methodology through an industrial case, and in particular validation of the compositional rules for some properties used in the partitioning decision process. MULTI PAR methodology is implemented as a tool, which is used in the validation process.
9.1 Introduction

Some of the latest hardware technologies support the realization of single chip “heterogeneous” platforms, consisting of different execution processing units - CPUs and FPGAs. On this type of platform application modules can be deployed either as software or hardware components, developed as, for example C/C++ and VHDL code, respectively. This capability challenges the designers to find the best configuration of the application into hardware (HW) and software (SW) components with respect to the requirements. This activity is known as HW/SW partitioning (or sometime abbreviated to partitioning) and it has been subject of study for over two decades (see for instance [1–3]).

In related literature (e.g. [4–6], among many others), the existing partitioning approaches provide solutions that are mostly based on the analysis of particular properties which are related to application runtime behavior, such as performance, energy consumption, memory footprint, and FPGA area. Even reduced to these parameters, finding optimal partitioning solutions is a difficult task. In addition, considering only a limited set of requirements conflicts with the growing complexity of new applications, where the respective development must satisfy a large number of technical requirements and constraints, as well as aspects related to project development, life cycle, etc. Applying the existing approaches has the disadvantage of providing partitioning solutions where dimensions such as maintainability, sustainability or upgradability are not taken into account. An initial question here is: besides the runtime related requirements, are there any other requirements or project development constraints that are of interest for carrying out partitioning decisions?

An answer to this question comes from a survey [7] highlighting that in addition to system requirements (such as performance, efficiency, recoverability, time behavior, reliability and security), the project development constraints (such as implementation, production, and production lead time) and the lifecycle requirements (e.g. maintenance costs, maintenance lead-time, upgradability, etc.) have a clear impact on the application partitioning.

One way to express the different types of requirements and project development constraints is through the Extra-Functional Properties (EFPs), i.e. properties that describe the quality characteristics of a component or a system. The energy consumption, worst case execution time, maintainability, portability, testing cost, design lead time are examples of this type of properties. In general, today there is a need to provide a way to consider more EFPs in the partitioning decision process, both at component and system levels. Taking into account many EFPs is not a trivial task. It requires large amount of effort
to measure or estimate the EFP values and to make a proper analysis of the dependencies between the EFPs in all possible system solutions. This effort exponentially increases with the number of components and it becomes practically impossible to measure the EFPs values for all component combinations.

An alternative to solve this problem is to use a bottom-up approach to calculate the system EFPs. Existing work [8] proposed rules able to analytically derive system EFPs from the involved components. These composition rules can be applied for calculating the system EFP values related to physical aspects (e.g. execution time, power consumption and footprint), lifecycle (e.g. upgradeability) and project development (e.g. overall development cost and implementation cost).

Considering this bottom-up approach a question of interest is: What is a partitioning decision process which takes into account multiple EFPs both at component and system levels?

A partial answer is provided by introducing a new partitioning methodology, called MULTI PAR [9, 10]. MULTI PAR obtains a solution by selecting existing components and using multi-criteria decision analysis (MCDA). The latter is a sub-discipline of operational research, that provides structured and transparent methods for breaking-down complex decision problems [11]. In addition, MULTI PAR combines component-based development (CBD) [12] and model-based design (MBD) [13], well-known approaches in software engineering. The CBD approach enables a formal definition of an embedded application as a set of deployable components into HW and SW units, and the reuse of the existing component. The MBD approach allows to perform platform-independent design in beginning of the design phase and postpones the platform-dependent design to a later stage.

In that version of MULTI PAR, called Component-MULTI PAR, the partitioning solution is achieved through a MCDA optimization which leads to the selection of the HW and SW components that meets the best EFPs values. However, at system level, where components interact with other components, more factors need to be considered, such as, the additional execution time related to the communication between the components or the additional costs required for performing system tests. The selection of the best partitioning solution, which satisfies the system EFPs of interest for partitioning, not only the ones at component level, is however an open challenge. In this paper we address this challenge by proposing an extension of MULTI PAR (called System-MULTI PAR). With this extension MULTI PAR is able to provide partitioning solutions with respect to system properties. Moreover, we present here a new developed tool, named Partitioning Design Tool (PDT) which implements the
9.2 MultIPAR Methodology

We start here by providing a short introduction of Component-MultIPAR, which allows selecting the best HW and SW component variants with respect to the EFPs of interest for a given application at component level - details in [9, 10]. We continue by describing the System-MultIPAR, which leads to a final partitioning solution by focusing on the analysis of the system EFPs. The illustration of the complete MultIPAR methodology is shown in Figure 9.1. The upper part of the process flow diagram is related to the Component-MultIPAR (the blue-bordered rectangle area), and the lower part is related to the System-MultIPAR (the green-bordered rectangle area).

9.2.1 Component-MultIPAR

MultIPAR considers a system as consisting of a platform (the heterogeneous platform) and an application, formally defined in a metamodel (see [9]). The application is architected as a set of interconnected component models, but the architectural design itself is not part of MultIPAR scope. MultIPAR adopts a component-based approach - a component is defined by its functional interface.
and by a set of EFPs. Different component implementations (in MULTIPAR terminology called variants) implement the same functionality but can have different EFPs. These variants can be HW variants (developed as, for example, VHDL code) or SW variants (developed as, for example, C++ code). Component variants may, however, vary from a non-functional perspective, that is, variants may have different EFP values. For instance, for intensive parallel

Figure 9.1: Complete MULTIPAR process flow - high level overview.
computing applications a HW variant will have a significantly better execution time than a SW one, but the situation regarding the maintainability values may be reversed.

In order to reduce the design space, MULTIPAR selects the best HW and SW variant for each component. As depicted in Figure 9.1 (see the Component-MULTIPAR part), the process starts with the identification of the EFPs relevant for partitioning from the application requirements and project constraints. The information related to the reusable components, as for instance the type (HW or SW variants) and the corresponding EFP values are retrieved from a repository of existing components (‘Component Repository’ in Figure 9.1). In case a component does not exist, the designer(s) will develop it. If the component is not implemented at the time of the evaluation MULTIPAR assumes that two possible implementations can be done, virtual variants: a HW and a SW variant, with estimated values for the EFPs of interest. At the end of the project, the information related to the newly designed components and their variants are supposed to be stored in the repository for possible reuse in future projects.

For all component variants, the EFP values are either measured or estimated. The variants with EFP values not satisfying the application requirements or project constraints are filtered out as candidates for the selection. For instance, there might exist a SW component variant which execution time is larger than the maximum allowed. As a consequence this variant will not be taken into account in the optimization. Subsequently, the prioritization of the EFPs is carried out, which means assigning a weight factor to each EFP.

The last activity in the process is the optimization, which is performed by applying a MCDA method. The outcome of this step provides a ranked list of HW and SW variants for each component.

Previous work [14] has shown the potential of using MCDA in partitioning processes. Consequently, in MULTIPAR the selection of the best HW and SW component is achieved via MCDA methods, able to deal with a finite number of alternative solutions. This type of methods belong to a specific class of MCDA, the Multiple Attribute Decision Making (MADM). For these methods, a typical problem is defined through the following elements:

- Objectives – representing the “reflections of the desires of the decision maker”, and “indicate the direction in which the decision maker wants the organization to work” [15]. In MULTIPAR the objectives are for instance the minimization of the system execution time or the minimization of the total development cost.

- Alternatives – representing the items to evaluate according to respective
attributes, in order to find the final solution (or set of solutions) [15]. In MULTI\textsc{par}, the alternatives represent the component variants.

- Criteria – express a measurement of effectiveness representing the basis for evaluation [16]. In MULTI\textsc{par} the criteria represent the EFP of interest for the partitioning.

- Attributes – define the characteristics, qualities or performance parameters of the alternatives. In MULTI\textsc{par} an attribute corresponds to the EFP value of an alternative.

- Preferences – reflect the decision maker opinions with respect to a given criteria. In MULTI\textsc{par} preferences are elicited via weight factors assigned by the decision maker(s).

In MADM methods the alternatives are explicitly given via a well-structured decision-making process, which by taking into account the stakeholders’ objectives and preferences allow to rank or classify the alternatives, or to choose a subset of alternatives [17]. MULTI\textsc{par} [14] applies the Evidential Reasoning (ER) [18] method, a ranking method able to address decision problems dealing with quantitative and qualitative criteria under uncertainties and randomness.

### 9.2.2 System-MULTI\textsc{par}

In this section, we describe the new part of MULTI\textsc{par} related to the system EFP analysis, defined by the process flow depicted in the lower area of Figure 9.1 (see System-MULTI\textsc{par}). The first part of MULTI\textsc{par} identifies which HW and SW variants were best satisfying the requirements and constraints expressed via component EFPs. From selection of best HW and SW components, there are $2^n$ possible system configurations (where $n$ is equal to the number of components) which would require $2^n$ measurements or estimations of system properties. This would be in practice impossible. For this reason, we here propose a process able to eliminate non-feasible configurations, analytically derive system composable EFPs from component EFPs, and select the components variants (HW or SW) that would fit the best to the system requirements and constraints, expressed as system EFPs.

This part of MULTI\textsc{par} builds on the system EFP composability rules provided in [8]. They allow to analytically derive some of the system EFP values from the EFP values of the involved components, as for example the system design cost, or the system memory footprint.
For each component, the first ranked HW and SW variants (see section 9.2.1) are combined into system variants, that is, possible HW/SW system deployment configurations. For instance, assume a system composed of 10 components, each represented by the identified two best ranked variants representing HW and SW technologies. Combining these variants to form the system, we end up with \(2^{10}\) possible system configurations. Here, our aim is to provide in a reliable and accurate manner a solution to select which, among all of these possible configurations would be the best with respect to the system requirements and development project constraints.

Below, we first introduce the relevant artifacts used in the process flow related to the System-MultiPar part in Figure 9.1), and then we describe each activity.

**Artifacts:**

- **System Requirements, Application Requirements, Project Constraints:** are a set of descriptions of the system and application characteristics and features, and project-related constraints, as for instance the maximum system execution time or project lead time.

- **System EFP Calculation Rules:** are the composition rules that can be used to derive system EFPs from the involved component EFPs, such as, the ones that we proposed in [8].

- **System Decision Matrix:** is a table where criteria and alternatives of the problem under analysis are collected. It includes the information such as a) the system EFPs relevant for performing the partitioning, and their priority values (i.e. weight values) which are, later on, needed for the optimization the system EFPs and b) all system variant combinations. For each system variant, the component variants that constitute it and the system EFP values have to be provided. A simple example of system decision matrix is shown in Figure 9.2.

**Activities:**

- **System EFP Identification:** the engineers involved in the partitioning process will identify the relevant EFPs to carry out the partitioning, based on the analysis of the system and application requirements, and the project management constraints. The selected EFPs will be entered into the system decision matrix. The system EFPs represent the decision criteria that will be used subsequently for the ranking of the system variants. In [7] we performed an analysis of EFPs which resulted into a
categorization of the EFPs with respect to the HW/SW partitioning and an assessment of their impact when performing partitioning decisions in the automation domain. This work can be used by engineers as guideline for the identification of EFPs of interest since many of the listed EFPs are common and relevant for other domains as well.

- **System Variant Configuration**: all system variant combinations will be generated based on the component optimization results and inserted into the system decision matrix. These system variants represent the candidates that will be optimized later on.

- **System EFP Value Assignment**: the system EFP values for each system variant are measured or estimated, after which they are inserted in the system decision matrix. A way to estimate some of the EFPs is proposed in [8]. For example the system design cost (SDC) might be calculated as:

\[
SDC = \sum_{i=1}^{n} CDC_{Ci} + EDC, \tag{9.1}
\]

where \(CDC_{Ci}\) is the component design time for the i-th component; \(EDC\) is the Extra Design Cost related to the design of the system architecture and the platform-related activities as for instance the cost of the effort required for configuring the communication between the HW and SW components; \(n\) is the total number of components.

However, there are system EFPs such as system safety or security, which values cannot be calculated from the component EFP values. In this case other estimation methods must be investigated.
9.3 MULTIPar Validation - Industrial Case

- **System Variant Filtering**: all system variants which do not satisfy the system requirements or the project management constraints will be filtered out. For example, a system variant might have a memory usage value that is far above the available memory of the given platform. Such variants are discarded as candidates for the optimization, and removed from the system decision matrix.

- **System EFP Prioritization**: a weight value is assigned to each system EFP in the system decision matrix, in order to carry out the optimization using MCDA methods. When several designers and project managers are involved in the decision process and they have to prioritize several EFPs, the prioritization might not be a trivial task. A way to facilitate the execution of this task might be given by the usage of existing structured methods. Examples of methods that might be used are the SMART method and its improvements SMARTS, and SMARTER [19, 20].

- **MCDA-based System Optimization**: the ER method is applied with the purpose of ranking the system variants that remain in the process. The highest-ranked variant will be the best partitioning solution to be deployed.

- **System Verification**: the proposed best partitioning solution will be verified against the requirements and constraints in order to check its correctness. In principle, in MULTIPar all requirements and constraints are supposed be given from the beginning. However there might be the risk that one or more requirements have not elicited from the beginning or have been modified during the development time. In this situation, the new or updated requirements or constraints have not been part of the analysis that led to the selected partitioning solution. If after the verification, the selected configuration does not meet these new/updated requirements or constraints, part of the MULTIPar process needs to be iterated.

9.3 **MULTIPar Validation - Industrial Case**

In this section we address the following research question: Is the proposed methodology feasible in an industrial context? In order to be able to answer this question, we first focus on demonstrating how the entire MULTIPar methodology was applied for partitioning an industrial application and checking if the
obtained solution was correct. Subsequently, we validate the composition rules proposed in [8] for the system execution time (SET), the system power consumption (SPC), the system memory footprint (SMF), and the system FPGA area utilization (SFPGAUAU) properties. For achieving this, we measure and calculate the EFP values of the system variant and then compute the accuracy of the calculated values with respect to the measured ones. This is followed by the ranking of the system variants based on the measured and calculated EFP values, in order to evaluate what is the impact on the solutions of the usage of the calculated or measured EFP values. Finally, we perform a sensitivity analysis in order to investigate the dependency of the system variant ranking on the weight variations. We conclude this section with a discussion on the obtained results.

9.3.1 Industrial project description

Below, a high-level overview of the system, the component-based application, the project setup, and a description of the repository of components related to this industrial case are provided.

Development Project

The development project used for the validation is of an “action research” type. The project team involved in the development of the application consisted of a project manager, a system architect with experience in SW design who has designed MULTIPAR, a HW designer, and two test engineers. The C code and VHDL code were automatically generated, consequently no developers were involved in this projects. A list with the main requirements and project constraints related to this industrial case is provided in Table 9.1.

System

The focus of the industrial case was on the realization of a safety-critical control system. A brushless direct current motor application was developed and deployed into a heterogeneous platform - Xilinx Zynq-7000 All Programmable SoC ZC702 (Zynq) [22]. This platform consists of two computational units, a dual-core CPU and an FPGA, where components can be deployed as executable C code or synthesized VHDL code. This platform is plugged into an Avnet motor control module [23], which is connected to the motor. Figure 9.3 illustrates a simplified block diagram related to the Avnet motor control module, the Zynq and the motor.
Table 9.1: List of key requirements or constraints

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Requirement or constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement 1</td>
<td>The minimum switching frequency for modulating the motor driver is 20kHz</td>
</tr>
<tr>
<td>Requirement 2</td>
<td>The maximum system execution time shall not exceed 30 microsec</td>
</tr>
<tr>
<td>Requirement 3</td>
<td>The power consumption shall not exceed 1W</td>
</tr>
<tr>
<td>Requirement 4</td>
<td>The application shall be field-upgradable</td>
</tr>
<tr>
<td>Constraint 1</td>
<td>The design phase shall be completed within 3 months</td>
</tr>
<tr>
<td>Constraint 2</td>
<td>The entire project shall be completed within 6 months</td>
</tr>
<tr>
<td>Constraint 3</td>
<td>The overall system development cost shall not exceed 100kUSD</td>
</tr>
<tr>
<td>Constraint 4</td>
<td>The application shall be deployed on the Zynq platform</td>
</tr>
<tr>
<td>Constraint 5</td>
<td>The application shall be developed and implemented by using the Mathworks Matlab and Simulink tools [21]</td>
</tr>
<tr>
<td>Constraint 6</td>
<td>The maximum outlay cost for the development environment shall not exceed 30kUSD</td>
</tr>
</tbody>
</table>

Application

The design of the application was based on a closed-loop feedback regulation of the speed where the three-phase pulse widths are modulated to control the motor driver. The speed regulation is performed using the hall effect sensors and the speed reference as input, then the calculated phase voltages as output to set the pulse width of each phase. A high level overview of the main application components is provided through the MathWorks Simulink [21] model shown in Figure 9.4. It consists of five main interconnected component models:

- the SPEED_DETECTOR which reads the hall effect sensor values, filters them and calculates the motor speed;
- the PI_CONTROLLER which calculates the speed error and the voltage references;
- the PWM_GENERATOR which generates the pulse-width modulated signals for steering the motor driver;
- the PHASE.STATE.MACHINE which regulates the commutation across the different phases
- the PHASE_DECODER which selects the correct pulse-width modulated sequence based on the current phases.

The controller also implements two safety protections: the safety-torque off (STO_PROTECTION) which allows the motor be safely coasted to a stop and the safely-limited speed (SLS_PROTECTION) which, when activated, prevents the motor from exceeding a pre-set speed limit.

![Figure 9.4: High-level overview of the application.](image)

With respect to the components described in Figure 9.4, the MathWorks Embedded Coder and the HDL Coder [21] were used to automatically generate different SW variants (C-code) and HW variants (VHDL-code). The component models were designed to generate fixed-point variants (C-code and VHDL-code) and floating-point variants (C-code). For the SW variants, different optimization parameters were set, which allowed the prioritization of either memory usage or execution time, such that for a given component more variants could be taken into account. Each component has three variants, two SW variants and one HW variant, with the exception of the SLS_PROTECTION and PICONTROLLER, with two SW and two HW variants each. All components and the related variants (including the implementation type, i.e. HW or SW) are listed in the table shown in Figure 9.5.
9.3.2 Applying MultiPar

Here, we describe how MultiPar is applied to achieve the partitioning of the motor control application, with the support of the PDT. This description is based on the flow activities depicted in Figure 9.1:

- **Modelling & Component Identification**: the components taken into consideration are the seven ones shown in Figure 9.4, and the 23 variants under analysis for the partitioning are shown in Figure 9.5.

- **Component EFP Identification & Value Assignment**: the project team built the list of the component EFPs most relevant for the partitioning based on the analysis of the requirements and constraints listed in Table 9.1. For instance, the component Execution Time properties was identified, as important, based on the analysis of the ‘Requirement
1’ and ‘Requirement 2’. The EFPs related to time (such as the Design Time, Testing Time, etc.) and cost (e.g. the Design Cost or the Design Lead Time) were derived from ‘Constraint 1’, ‘Constraint 2’ and ‘Constraint 3’. The identified EFPs are listed in Table 9.2, which includes the name properties and their identifiers (see first and second columns).

In Figure 9.6 the component decision matrix is depicted. It contains the component names and the identifier (ID) of the related HW and SW variants (see first and second columns) and the identified EFPs (see columns 3-13).

<table>
<thead>
<tr>
<th>Component EFP</th>
<th>Identifier</th>
<th>Weight Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time</td>
<td>ET</td>
<td>100</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>PC</td>
<td>90</td>
</tr>
<tr>
<td>Design Lead Time</td>
<td>DLT</td>
<td>90</td>
</tr>
<tr>
<td>Implementation Lead Time</td>
<td>ILT</td>
<td>80</td>
</tr>
<tr>
<td>Testing Lead Time</td>
<td>TLT</td>
<td>75</td>
</tr>
<tr>
<td>Requirement Elicitation Lead Time</td>
<td>RELT</td>
<td>60</td>
</tr>
<tr>
<td>FPGA Area Utilization</td>
<td>FPGAAU</td>
<td>55</td>
</tr>
<tr>
<td>Memory Footprint</td>
<td>MF</td>
<td>50</td>
</tr>
<tr>
<td>Design Cost</td>
<td>DC</td>
<td>45</td>
</tr>
<tr>
<td>Testing Cost</td>
<td>TC</td>
<td>45</td>
</tr>
<tr>
<td>Requirement Elicitation Cost</td>
<td>REC</td>
<td>40</td>
</tr>
<tr>
<td>Implementation Cost</td>
<td>IC</td>
<td>30</td>
</tr>
<tr>
<td>Traceability</td>
<td>TR</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 9.2: Identified Component EFPs and related weight values.

To assign values to the component variant EFPs, several methods were applied. The values related to the project development, as for instance the DLT, TLT, TLT, RELT, DC, TC, and REC properties were estimated by the members of the project team based on their experience. The TR values were derived based on the existence of a link between the component model and the C/VHDL code. The values of properties such as the ET, PC, FPGAAU, and MF were measured. The measuring approaches used for the ET, PC, FPGAAU and MF are described in [24]. All EFP values are reported in Figure 9.6.

- Component Filtering: the variant EFP values were verified against the constraints in order to evaluate if any system variants need to be filtered out. For instance, it was verified if the estimated design time of each component was not exceeding the maximum allowed design time.
<table>
<thead>
<tr>
<th>System</th>
<th>Variant ID</th>
<th>Variants</th>
<th>SW/ HW</th>
<th>EFP</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 9.6:** Component Decision Matrix with ranking values and selection of the best HW and SW variants for each component included.
planned for the given component. All component variants were assessed as suitable candidates for the component MCDA optimization. Consequently none of them were discarded at this stage.

- **Component EFP Prioritization**: A way to assign the weights is by the definition of a scale varying from 0 (not important) to 100 (extremely important), and then normalize the assigned values in the range of $[0, 1]$ [25]. Consequently, all project members were involved in the operation of eliciting the EFP weight values by rating the importance of the EFPs, from 0 to 1 using a step of 5 or 10. For instance at component level, the ET was considered to have the highest importance because if the ‘Requirement 1’ is not fulfilled the motor might not be controlled. Higher importance was also assigned to the DLT, since ‘Constraint 1’ was considered of key importance by the project manager. In general, the ET and PC were considered to have an high importance from the HW and SW engineer perspective, while the DLT or ILT were considered the ones with highest priority from the management perspective. The assigned weight values are shown in Table 9.2. The normalized value are, instead, reported in the first row of the component decision matrix (see the ‘Normalized Weight’ column). Figure 9.7 shows how the weight factors for the component EFPs are set in the PDT.

![Figure 9.7: PDT - Component EFPs and related weight values.](image-url)
• **MCDA-based Component Optimization**: here, the ranking of the best HW and SW variants for each component was performed by using the ER method. For each component, the ranking values are reported in the ‘ER Ranking’ column in Figure 9.6, while the highest ranked HW and SW variants for each component are indicated in the last two columns. In Appendix 9.7 a detailed description of how the ER method is applied in **MULTIPAR** is provided. A screenshot of PDT displaying the ranking for the PHASE_DECODER component is shown in Figure 9.8.

![Component Decision Matrix](image)

**Component Name**

**Ranking values**

**Best HW and SW variants**

Figure 9.8: PDT - PHASE_DECODER (C6) component variant ranking.

Out of this activity 14 component variants were selected for the next optimization, i.e. the system EFP phase.

• **System Variant Configuration**: all possible combinations, that is, system variants were generated. When considering 7 components and 2 associated best HW and SW component variants, the number of all possible combinations is $2^7$. Hence, there are 128 system variants to be considered for the partitioning analysis at system level.

• **System EFP Identification** and **System EFP prioritization**: the project team selected the relevant system EFPs and prioritized them, based on analysis of the system requirements, project constraints (see Table 9.1)
and the available component EFPs. For instance, the 'Constraint 1' brings to the identification of the Total Development Lead Time as of one properties influencing the partitioning decisions. The selected EFPs are listed in Table 9.3. Subsequently, in a similar way as it was performed for assigning the component EFP weights, the project team assigned the system weight factors to the selected properties. Specifically, each weight was supposed to have a value in the range of [0,100] with a step of 5 or 10, and later one all of the assigned weights values were normalized between 0 and 1. At system level, the highest importance was given to the Total Development Lead Time by the project manager, while the lowest importance was assigned to the System Development Environment Cost, since most of the required tools were already available from previous project executions.

<table>
<thead>
<tr>
<th>System EFP</th>
<th>Identifier</th>
<th>Weight Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Execution time</td>
<td>SET</td>
<td>70</td>
</tr>
<tr>
<td>System Power Consumption</td>
<td>SPC</td>
<td>60</td>
</tr>
<tr>
<td>System Memory Footprint</td>
<td>SMF</td>
<td>50</td>
</tr>
<tr>
<td>System FPGA Area Utilization</td>
<td>SFPGAAU</td>
<td>40</td>
</tr>
<tr>
<td>System Development Environment Cost</td>
<td>SDEC</td>
<td>30</td>
</tr>
<tr>
<td>Total Development Cost</td>
<td>TDC</td>
<td>80</td>
</tr>
<tr>
<td>Total Development Lead Time</td>
<td>TDLT</td>
<td>100</td>
</tr>
<tr>
<td>System Upgradability</td>
<td>SUP</td>
<td>90</td>
</tr>
</tbody>
</table>

Table 9.3: Identified System EFPs and related weight values.

- **System EFP Value Assignment**: the assignment of the system EFP values for all 128 configurations was carried out. For properties such as SDEC, TDC, TDLT and SUP the values were calculated. The SET, SPC, SMF and SFPGAAU values were either measured or calculated used the composition rule proposed in [8]. The measured values were obtained by using the methods described in [24]. Examples of system variants and related EFP values can be found in Figure 9.9, representing the system decision matrix.

- **System Variant Filtering**: here the values of the system EFPs are checked against the requirements and constraints. A key constraint related to the minimum switching frequency (see 'Requirement 1' in Table 9.1) required the filtering out of the SW variants of PWM_GENERATOR, the PHASE_DECODER and the STO_PROTECTION components. This operation resulted in a further reduction of the design space, conse-
Figure 9.9: System Decision Matrix for the 16 suitable system variants, with the ranking values included.
sequently only 16 system variants were considered suitable to be part of the next activity, i.e. the optimization.

- **MCDA-based System Optimization**: the ranking of all of the system variants was performed during this activity. The ER method was applied. The ranking of the 16 system variants is shown in Figure 9.9. The system variants with measured SET, SPC, SMF and SFPGAAU EFps are indicated by the blue-light colour in the first column of the decision matrix shown in Figure 9.9. The ranking of system variants was performed via the PDT. A screenshot showing the decision matrix and the ranking of the 16 variants is provided in Figure 9.10. The highest ranked system variant is the one with ID equal to 80.

![System Decision Matrix](image)

**Figure 9.10: PDT - System variant ranking.**

- **System Verification**: the test engineers deployed the selected configuration on the Zynq. The motor was shown to be properly controlled and
all of the required functions were working correctly. Consequently, no further iterations of MULTIPar were considered necessary by the project team.

9.3.3 Validating MULTIPar partitioning solutions

This subsection is focused on validating the partitioning solution(s) obtained using MULTIPar with respect to its accuracy for the given industrial case. It is divided into several parts. First we present the results related to the accuracy of the systems EFPs values with respect to the following properties: SET, SPC, SMF and SFPGAU. Then, we present the results related to the system variant ranking with respect to measured and calculated EFP values. Finally, we present a sensitivity analysis performed in relation to weight variations.

System EFP values

This part of our validation strategy was focused on obtaining results to analyze the accuracy of the rules proposed in [8] for properties that were measurable on the Zynq. Specifically, we analysed the following rules for:

- System Execution Time (SET): which is calculated used Formula 9.2 and Formula 9.3, depending on the execution between two components (sequential or parallel), and the type of communication channels (i.e. SW-SW, SW-HW, HW-SW, and HW-HW), details can be found in [8],

\[
SRT = ET_{Ca} + ET_{Cb} + CT_{Ca,Cb} \tag{9.2}
\]

\[
SRT = \max(ET_{Ca}, ET_{Cb}) + CT_{Ca,Cb} \tag{9.3}
\]

where based on the control flow, \(ET_{Ca}\) is the execution time of the first component and \(ET_{Cb}\) is the one related to the second component, while \(CT_{Ca,Cb}\) is the additional execution time required for data communication (e.g. considering the Zynq, an example of additional time is the one required for two components send and receive data via the AXI bus [26]).

- System Power Consumption (SPC): which is computed according to Formula 9.4
\[ SPC = \sum_{i=1}^{n} CPC_{Ci} + EPC \]  \hspace{1cm} (9.4)

where the \( CPC_{Ci} \) is the component power consumption for the \( i \)-th component; \( EPC \) is the additional power consumption related to the system architecture and platform (e.g. the static power consumption of the Zynq); \( n \) is the total number of components.

- **System Memory Footprint (SMF)**: which is calculated according to Formula 9.5

\[ SMF = \sum_{i=1}^{n} CMF_{Ci} + EM \]  \hspace{1cm} (9.5)

where the \( CMF_{Ci} \) is the component static memory for the \( i \)-th component; \( EM \) is the additional static memory utilization (e.g. the amount of additional static memory used for performing write/read operations in the AXI registers); \( n \) is the total number of components.

- **System FPGA Area Utilization (SFPGA AU)**: which is derived by using Formula 9.6

\[ SFPGA AU = \sum_{i=1}^{n} CFA_{Ci} + EAU, \]  \hspace{1cm} (9.6)

where the \( CFA_{Ci} \) is the FPGA area utilization for the \( i \)-th component; \( EA \) is the additional FPGA area utilization as for instance the extra look-up table (LUTs) corresponding to the base area; \( n \) is the total number of components.

With respect to Table 9.3 the properties whose values where estimated, specifically the TDC, TDLT, UP, and SDEC were not part of this analysis. We measured the values of the SET, SPC, SMF and SFPGA AU properties at component level and system level, and then we verify whether the values of the system EFP calculated by applying the proposed rules provided results that are close to the real measurements. To achieve this, we performed the measurements of the EFPs under analysis, according to the methods described in [24]. The measured EFP values for the components are shown in Figure 9.6.
MultiPar Validation - Industrial Case

For all component variants we measured the ET, PC, while the MF and the FPGAAU were respectively measured for the SW variants and the HW variants.

After that all 128 system configurations were generated, we analyzed them in order to extract a significant and representative subset of system variants for validating the formulas. The extraction process was based on the analysis of the number of communications between HW and SW computing units, i.e. from the FPGA to the CPU (HW-SW) and vice versa (SW-HW), for each system configuration. This analysis leads to cluster all 128 configurations into 8 groups. For instance, the first group was containing all system configurations with zero HW-SW/SW-HW communication channels (i.e. the system variant with all SW component variants and the one with all HW component variants) while the second group was containing all system configurations which had one HW-SW/SW-HW communication channel, and so forth.

In each group, system variants differ for the number HW or SW component variants. Each system variant having the same ratio, i.e.

\[
\text{ratio} = \frac{\text{number}_{\text{HW component}}}{\text{number}_{\text{SW component}}} \tag{9.7}
\]

has been assumed to have a similar runtime behavior. Consequently, for each different ratio value, only one representative (and its complementary) has been selected from each group. At the end of the extraction process we obtained 34 system variants. The selected system variants are shown in Figure 9.12, where

- the first column is used as unique identifier of the selected 34 variants.
- the second column contains the selected system variants. Each system variant is represented by an unique identifier (ID). These identifiers were assigned to all 128 system configurations under analysis. When referring to a system variant by using this identifier, the system variant will be called as follows, i.e. ID-system variant. For instance, the system variant with ID equal to 122, will be referred as the 122-system variant.
- the column 2-8 include the component variant types for each variant, i.e. HW or SW.
- the 8 groups are shown in column 9. For instance, the group number ‘1’ includes a system variant which has one HW-SW communication channel (see 2-system variant), and its complementary which has one
SW-HW communication channel (see 122-system variant). The communication channels between the component variants were based on the topology described in Figure 9.11, for instance, the 122-system variant has only one communication channel from SW to HW, between the component C6 and C7.

- in the last two columns, the number of component HW variants and SW variants for each system variant are shown, which ratio has been used to select just one representative from all of the system variants having the same ratio in a given group.

The measured and calculated values of the properties under analysis for the 34 selected system variants are shown in Figure 9.13, where for each EFP the measured values are shown in the column named ‘M’ while the calculated ones are shown in the column named ‘C’.

\[
acc = 1 - \frac{|M_i - C_i|}{M_i}
\]  

(9.8)

where \(M_i\) corresponds to the measured EFP value of system variant \(i\) and \(C_i\) is the calculated EFP value for the given variant \(i\).

For each EFP the corresponding accuracy of each alternative is shown in Figure 9.13, in the column named ‘A’.

Moreover, for each EFP we computed the normal distribution [27] using the relative error values. For all of the four EFPs, the mean values, the standard
deviations and the distributions have been calculated according the formulas 9.9, 9.10, and 9.11.

\[
G(x) = \frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}
\]  
(9.9)
Figure 9.14: System Variant EFP values for the SET, SPC, SMF, and SFPGAAU: measured (M) and calculated (C), and accuracy (A).

\[
\mu = \frac{1}{N} \sum_{i} |x_i| \quad (9.10)
\]

\[
\sigma = \sqrt{\frac{1}{N} \sum_{i} (|x_i| - \mu)^2} \quad (9.11)
\]

In Figure 9.14 the normal distribution curves for the SET, SPC, SMF, and SFPGAAU are plotted in the same graph, in order to facilitate the comparison. The normal distribution of each property have been normalized.
System ranking values

Here, our work focused on analyzing the ranking position achieved by using the calculated EFP values of the SET, SPC, SMF and SFPGAAU with respect to the ranking achieved by using the measured values for the same properties. Both ranking operations were performed using the ER method and by applying the same EFP weights. All of the EFPs and related weights listed in Table 9.3 were part of the analysis. The values of the SDEC, TDC, TDLT, and SUP properties were analytically derived using the component EFP values shown in Figure 9.6 and the composition rules provided in [8]. Two ranking operations were performed. First, in the first operation the values of the SDEC, TDC, TDLT, and SUP EFPs were used together with the measured values of the SET, SPC, SMF, and SFPGAAU properties. Subsequently, they were used in combination with the calculated values of the SET, SPC, SMF and SFPGAAU properties. The measured and calculated values of the SET, SPC, SMF, and SFPGAAU EFPs are shown in Figure 9.13, while the values of the SDEC, TDC, TDLT, and SUP properties are shown in Figure 9.9. When performing the ranking operations, all of the 34 selected system variants (see Figure 9.12) were used. In Figure 9.15 the ranking position of the system variants for both cases are shown.

In Figure 9.15, the system variants that have the same ranking position, for
<table>
<thead>
<tr>
<th>RANKING POSITION</th>
<th>SYSTEM VARIANT ID (MEASURED EFP VALUES)</th>
<th>SYSTEM VARIANT ID (CALCULATED EFP VALUES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>128</td>
<td>126</td>
</tr>
<tr>
<td>2</td>
<td>126</td>
<td>128</td>
</tr>
<tr>
<td>3</td>
<td>111</td>
<td>111</td>
</tr>
<tr>
<td>4</td>
<td>101</td>
<td>101</td>
</tr>
<tr>
<td>5</td>
<td>124</td>
<td>124</td>
</tr>
<tr>
<td>6</td>
<td>84</td>
<td>84</td>
</tr>
<tr>
<td>7</td>
<td>122</td>
<td>115</td>
</tr>
<tr>
<td>8</td>
<td>115</td>
<td>122</td>
</tr>
<tr>
<td>9</td>
<td>127</td>
<td>127</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>11</td>
<td>120</td>
<td>120</td>
</tr>
<tr>
<td>12</td>
<td>119</td>
<td>119</td>
</tr>
<tr>
<td>13</td>
<td>88</td>
<td>88</td>
</tr>
<tr>
<td>14</td>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>15</td>
<td>118</td>
<td>118</td>
</tr>
<tr>
<td>16</td>
<td>80</td>
<td>19</td>
</tr>
<tr>
<td>17</td>
<td>19</td>
<td>80</td>
</tr>
<tr>
<td>18</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>19</td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>20</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>21</td>
<td>18</td>
<td>91</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>24</td>
<td>2</td>
<td>31</td>
</tr>
<tr>
<td>25</td>
<td>97</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>7</td>
<td>97</td>
</tr>
<tr>
<td>28</td>
<td>22</td>
<td>7</td>
</tr>
<tr>
<td>29</td>
<td>24</td>
<td>22</td>
</tr>
<tr>
<td>30</td>
<td>91</td>
<td>24</td>
</tr>
<tr>
<td>31</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>32</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>33</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>34</td>
<td>23</td>
<td>23</td>
</tr>
</tbody>
</table>

Legend

<table>
<thead>
<tr>
<th></th>
<th>Same ranking position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Different ranking position</td>
</tr>
</tbody>
</table>

Figure 9.15: ER Ranking Position Comparison between system variants with measured EFP values and system variants with calculated EFP values.

...both cases, are blue-highlighted.

When applying ER, a ranking value is assigned to each alternative. Conse-
quently, we were interested in performing an analysis focused on the ranking values of the same system variant with respect to the calculated and measured EFP values. In the same way as for the EFP values, we calculated the accuracy of ER configuration values according to Formula 9.12.

\[
\text{acc}_{\text{ER, ranking value}} = 1 - \frac{|RV_{M_i} - RV_{C_i}|}{RV_{M_i}} \quad (9.12)
\]

where \(RV_{M_i}\) corresponds to the ranking value of the system variant \(i\) where the EFP values were measured; \(RV_{C_i}\) is the ranking value of the same system variant \(i\) with calculated EFP values. The results are shown Figure 9.16, where the first column includes the system variant ID, while given a system variant the second and the third columns show the corresponding ER ranking values, respectively for the measured and calculated EFP values.

**Ranking Sensitivity Analysis**

As discussed in [25], analysis to investigate wherever conclusions are sensitive to changes gives the possibility to explore the effects of decision makers about values/priorities, or a different perspective of the problem. In MCDA the weight prioritization influences the final results. Consequently, our aim, here, was to evaluate the sensitivity of the system variant ranking values when the weight of a given EFP is varying. For this analysis, we considered the SET and the SUP properties. This choice was based on the work presented in [7], which showed that experts consider the SET to be prevalently satisfied by a HW implementation, and that a highest SUP is easier to be achieved by a SW implementation. The analysis was performed on the 16 system variants shown in Figure 9.9. Specifically, we changed the SET and SUP weights, from 0 to 100, by using a factor of 10. For each properties, we plotted a graph showing how the ranking value of each system configuration varies when the weight of a given EFP changes (see Figure 9.17 for the SET property and 9.18 for the SUP property). For each figure, the values of the EFP weights are reported in the x-axis, while the y-axis represents the ranking values of each variant. The grey vertical line indicates the nominal weight value, which are respectively 70 for the SET and 90 for the SUP (see Table 9.3). In addition, to analyse the sensitivity of ranking value in relation to the weight variation, we computed the first derivative at the nominal weight value for both the SUP and the SET properties. The results are shown in Figure 9.19 through a histogram graph, which includes the values of the derivatives of the SUP and SET for each alternative with respect to both criteria.
<table>
<thead>
<tr>
<th>SYSTEM VARIANT ID</th>
<th>RANKING (MEASURED EFP VALUES)</th>
<th>RANKING (CALCULATED EFP VALUES)</th>
<th>ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.484</td>
<td>0.480</td>
<td>0.992</td>
</tr>
<tr>
<td>2</td>
<td>0.487</td>
<td>0.488</td>
<td>1.000</td>
</tr>
<tr>
<td>3</td>
<td>0.496</td>
<td>0.492</td>
<td>0.991</td>
</tr>
<tr>
<td>5</td>
<td>0.477</td>
<td>0.471</td>
<td>0.998</td>
</tr>
<tr>
<td>10</td>
<td>0.447</td>
<td>0.444</td>
<td>0.992</td>
</tr>
<tr>
<td>15</td>
<td>0.497</td>
<td>0.488</td>
<td>0.983</td>
</tr>
<tr>
<td>19</td>
<td>0.533</td>
<td>0.531</td>
<td>0.997</td>
</tr>
<tr>
<td>22</td>
<td>0.468</td>
<td>0.463</td>
<td>0.991</td>
</tr>
<tr>
<td>23</td>
<td>0.416</td>
<td>0.411</td>
<td>0.986</td>
</tr>
<tr>
<td>24</td>
<td>0.467</td>
<td>0.457</td>
<td>0.980</td>
</tr>
<tr>
<td>30</td>
<td>0.509</td>
<td>0.503</td>
<td>0.993</td>
</tr>
<tr>
<td>31</td>
<td>0.490</td>
<td>0.488</td>
<td>0.995</td>
</tr>
<tr>
<td>35</td>
<td>0.511</td>
<td>0.506</td>
<td>0.999</td>
</tr>
<tr>
<td>40</td>
<td>0.526</td>
<td>0.519</td>
<td>0.995</td>
</tr>
<tr>
<td>43</td>
<td>0.514</td>
<td>0.509</td>
<td>0.991</td>
</tr>
<tr>
<td>48</td>
<td>0.448</td>
<td>0.441</td>
<td>0.989</td>
</tr>
<tr>
<td>80</td>
<td>0.533</td>
<td>0.529</td>
<td>0.994</td>
</tr>
<tr>
<td>84</td>
<td>0.614</td>
<td>0.610</td>
<td>0.994</td>
</tr>
<tr>
<td>88</td>
<td>0.573</td>
<td>0.569</td>
<td>0.994</td>
</tr>
<tr>
<td>91</td>
<td>0.451</td>
<td>0.450</td>
<td>0.995</td>
</tr>
<tr>
<td>97</td>
<td>0.486</td>
<td>0.480</td>
<td>0.999</td>
</tr>
<tr>
<td>99</td>
<td>0.562</td>
<td>0.559</td>
<td>0.994</td>
</tr>
<tr>
<td>101</td>
<td>0.640</td>
<td>0.638</td>
<td>0.994</td>
</tr>
<tr>
<td>111</td>
<td>0.653</td>
<td>0.650</td>
<td>0.999</td>
</tr>
<tr>
<td>115</td>
<td>0.608</td>
<td>0.605</td>
<td>0.996</td>
</tr>
<tr>
<td>118</td>
<td>0.546</td>
<td>0.544</td>
<td>0.997</td>
</tr>
<tr>
<td>119</td>
<td>0.562</td>
<td>0.556</td>
<td>0.994</td>
</tr>
<tr>
<td>120</td>
<td>0.589</td>
<td>0.581</td>
<td>0.999</td>
</tr>
<tr>
<td>122</td>
<td>0.608</td>
<td>0.604</td>
<td>0.994</td>
</tr>
<tr>
<td>124</td>
<td>0.622</td>
<td>0.621</td>
<td>0.997</td>
</tr>
<tr>
<td>125</td>
<td>0.662</td>
<td>0.661</td>
<td>0.997</td>
</tr>
<tr>
<td>127</td>
<td>0.600</td>
<td>0.598</td>
<td>0.994</td>
</tr>
<tr>
<td>128</td>
<td>0.671</td>
<td>0.657</td>
<td>0.980</td>
</tr>
</tbody>
</table>

Figure 9.16: ER ranking value comparison between system variants with measured EFP values and system variants with calculated EFP values.

9.3.4 Result Discussion

Our aim was to provide an answer to the question: *Is the proposed methodology feasible in an industrial context?*

By the work presented in Section 9.3.2, we demonstrated how the entire MULTIPAR process can be applied step-by-step for partitioning a control industrial application. The solution achieved was deployed on the given platform, and was able to properly control the motor. This solution was satisfying the ini-
tial system requirements and project constraints, as for instance the minimum required switching frequency or the maximum available design lead time. For the application described in Figure 9.4, we considered 128 system configurations and eight EFPs, which size-wise was manageable in terms of component
Figure 9.19: SUP and SET variation comparison.

EFPs measurement effort, weight setting, and computations.
As part of our validation strategy we also aimed to analyze how accurate are the MultiPar solutions when applying the composition rules proposed in [8] to obtain system EFPs. In particular, we analysed the SET, SPC, SMF and SFPGA AU. Specifically, we compared the measured values of these properties with the calculated ones, which were analytically derived by using the formulas proposed in [8]. For each property, we computed the accuracy, which results were between 0.958 and 0.0990. In addition, in Figure 9.14 we presented the normal distribution curves for all four EFPs. As general consideration lower standard deviations (the highest one is 0.0390) correspond to deterministic calculation errors (lower uncertainties), which makes it possible to be compensated (as a future refinement) using just offsets.

The analysis of the ranking position achieved by using the measured and calculated EFPs for the SET, SPC, SM, and SFPGA AU EFPs, has shown that 19 out of 34 system variants (i.e. 55.8%) have the same ranking position (see Figure 9.15). The different positions of the remaining 15 system variants is due to the fact the calculated EFPs values of these alternatives have a lower accuracy. This is, for instance, the case of the 91-system variant or 128-system variant, which have respectively the lowest accuracy for the SPC and the SET properties. However, with respect to the analysis performed on the ER ranking values of the configurations (see Figure 9.16), it can be seen that for the same system variants the accuracy is equal or higher than 0.980 for 33 out of 34 system variants, with an exception for the 91-system variant, which accuracy is equal to 0.886.

The high accuracy of the composition rules implies that the proposed bottom-up approach, i.e. starting from component EFPs and analytically deriving the system EFPs, can be used to minimize the engineering effort required to measure all possible system variant configurations.

In addition, we performed the sensitivity analysis, (see Section 9.3.3), to understand the impact of the SET and SUP weight values on the variant ranking. With respect to Figure 9.17 and 9.18, we analyzed the behavior of two relevant configurations: the 80-system variant (which is the one that we deployed in the Zynq) and 1-system variant (which is fully HW-component variant implemented - see Figure 9.12). With respect to the variation of the SET property, the 80-system variant is always the highest ranked, but the ranking position of the 1-system variant is increasing from position 14 (i.e. when the weight value is equal to 0) to the position 3 (i.e. when the weight value is equal to 100). This is due to the fact that the SET values of the 1-system variant is the best one among all 16 alternatives. When considering the SUP property, the fully HW variant is preferred to the 80-system variant, when the value of weight is
equal to 20, but the situation is inverted when the importance of the SUP starts
to increase. With a weight value ranging from 30 to 100, the 80-system variant
is always ranked as the first one, while at maximum weight value (e.g. 100) the
1-system variant reaches the 10-th position. In Figure 9.19 we compared the
behavior of 16 alternatives with respect to the variation of the nominal weight.
With respect to the SET weight value, it shows that the rate of the change
of the ranking value is larger for the 1-system variant, 24-system variant and
80-system variant, and this is due to the lower values of the SET. Viceversa,
variants such as 27-system variant or 51-system variant are not affected by the
SET value variation due to the higher value of execution time. Compared to
the SET property, the sensitivity is lower for the SUP one. However, variants
such as the fully-HW one and the 3-system variant are more impacted by the
SUP weight value variation than the remaining ones. As general consideration,
this analysis has shown that EFP weight assignment plays a key role on the
selection of the final solution to deploy on a given platform.

9.4 Conclusion and Future Work

Considering many system EFPs when performing partitioning is challenging
because it requires to measure/estimate the EFPs values and to analyse their
dependencies in relation to all possible system configurations. In this paper,
we proposed an extension of MULTIpar, a novel methodology for partition-
ing embedded applications, which allows to achieve solutions with respect to
multiple properties at system level. By combining MCDA, CBD and MBD,
and using a bottom-up approach (i.e. deriving system properties from com-
ponent EFPs) MULTIpar facilitates the analysis of many system EFPs and
their dependencies and allows to rank all of the possible system configurations.
Moreover, we presented a tool which implements MULTIpar. Through an in-
dustrial case, we demonstrated that MULTIpar is suitable for being applied in
an industrial context and proved that MULTIpar provides reliable deployable
solutions. In particular, for the given application we showed that the solutions
achieved by using MULTIpar in combination with the composition rules pro-
posed in [8] are sufficiently close to the solutions that used measured values of
EFPs.

However, MULTIpar has some challenging points which can be conside-
red for future research directions.

The first challenge is related to the non-composable EFPs, as for instance
system security or safety. Today, the engineers applying MULTIpar have to
specify/calculate this type of system EFPs manually for each possible system configuration. This approach is quite infeasible when a large number of system configurations has to be taken into account.

The second challenging point is related to the way to assign the weight values. This is a crucial task for determining the proper system configuration to deploy on the platform. In MULTIPAR, this task is performed by the engineers and managers that are involved in the process, and we suggested a few methods (e.g. SMART) to support them when assigning the weights. Nevertheless, we envisage the need to complement MULTIPAR with additional activities that explicitly and in a structured way allow to take into account all stakeholders’ concerns and guide them when performing the EFP prioritization.

However, for many EFPs the partitioning decisions do not have any or significant impact [7], which makes the method easier to apply.

A third consideration is related to the number of system variants that can be generated and taken into account in MULTIPAR analysis. Assuming to have the best HW and SW variants for \( k \) components, the number possible alternatives is equal to \( 2^k \). In this scenario, we see the need to enhance MULTIPAR with a solution able to decrease the design space exploration.

Last but not least, it might be of interested to extend the analysis performed with respect to the automation domain in [7] to other domains such as automotive and avionics domains.

9.5 Acknowledgments

One of the authors, Gaetana Sapienza, wants to acknowledge the support of KKS foundation through ITS-EASY, an Industrial Research School in Embedded Software and Systems and of the Swedish Foundation for Strategic Research through the RALF3 project, and would like to thank Roger Jansson.

References


9.6 Appendix: PDT - MultiPar Tool

To partition an application using MultiPar today, several existing tools can be used. However, these tools implement only one activity of the process flow described in Figure 9.1. As a consequence, more tools need to be used to execute the entire MultiPar methodology. Some examples are the Intelligent Decision System (IDS) tool [28], Visual PROMETHEE [29] and SANNA [30] which support decision makers in solving decision making problems. These tools might be combined with Microsoft Excel [31] for calculating the EFP values. However, many MultiPar activities require some manual effort such as, for instance, the generation of system variant combinations, filtering operation of the variants, the handling of the components, their variants and related EFPs in a repository. In order to provide a support to the engineers who want to apply MultiPar, we have developed Partitioning Design Tool (PDT). The latter implements the main MultiPar features and provides the possibility of...
performing the partitioning by following step-by-step the entire process flow described in Section 9.2 and by Figure 9.1.

The PDT has been implemented by using the following development environment:

- PyCharm [32] for implementation and debugging
- SQLiteManager [33] for managing the repository.

The PDT was developed by applying component-based and object-oriented approaches. For the developing PDT, some design patterns have been applied such as the Model-View-Controller [34]. A high-level overview of the software architecture of PDT is provided in Figure 9.20. This figure shows the main components and the operations that they implement. A component offers a set of services that allows the management of a specific resource (e.g. the Component Repository, Component Decision Matrix, System Decision Matrix) or a set of resources. The core component is the Model component, which handles the functionalities provided by all of the other components related to the system, components, algorithms, etc.

![Figure 9.20: PDT - high level component-based architecture.](image-url)

The DataModel allows the management of information related to the component and system decision matrices such as, for instance, the associated EFPs. The SystemModel describes the system structure (i.e. the interconnected components, and related variants). In particular, it enables the handling of the system variants and the component variant objects, and reads the system EFPs.
and the related values. It also allows the management of the operation related to the component and system EFP weights. The \textit{DBInterface} provides methods for reading, updating and inserting data into the repository. It allows the reading of the values from the database without using queries in order to decouple the database complexity from the code. The \textit{AlgorithmInterface} exposes the interface for executing the algorithms (e.g. ER) and reads the results. The \textit{ComponentViewController} and \textit{SystemViewController} are the components that enable the interactions with the user (e.g. component and system filtering operations, exporting of the component EFPs values and importing of the calculated system EFPs) and they manage how results are displayed to the user. The \textit{ComponentTableModel} and \textit{SystemTableModel} are components which handle the component and system decision matrices.

The main functionalities that PDT implements are listed in Table 9.4.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Short Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database handling</td>
<td>The user can operate on the repository to add, remove or delete components and system variants</td>
</tr>
<tr>
<td>Components and system handling</td>
<td>The user can create a new project, select the components from the database, import the components in the project and set the system topology</td>
</tr>
<tr>
<td>Component variants filtering</td>
<td>The user can filter the component variants. Different filters can be defined. Less than, greater than and equal to are examples of filtering operations that can be used to remove variants that do not meet the system requirements or the project constraints</td>
</tr>
<tr>
<td>System variants filtering</td>
<td>The same filtering operations described for the component can be performed at system level</td>
</tr>
</tbody>
</table>
PDT implements two different ways for calculating the system EFPs: a) system EFP values that are not related to the topology (e.g. design cost) can be directly calculated using some pre-implemented functions in PDT, b) system EFP values that are related to the topology (e.g. the worst case execution time) can be calculated using a preformatted and automatic generated Matlab script. The user can use the MathWorks Matlab tool [21] script and the component EFP value data (which can be exported from the PDT) in order to calculate the system EFP values. When all of the system EFP values are calculated (using for instance Matlab or another tool) the user can export the values into a CSV file.

A CSV file containing all of the calculated system EFP values can be imported into the PDT and later on be used for the ranking.

The system and component EFP weights and the related utility function preferences can be set and will be used for the ranking.

The ranking of the component variants can be done by using the ER algorithm, which is implemented by the PDT.

The ranking of the system variants can be done by using the ER algorithm, which is implemented by the PDT.

| System EFPs values calculation | PDT implements two different ways for calculating the system EFPs: a) system EFP values that are not related to the topology (e.g. design cost) can be directly calculated using some pre-implemented functions in PDT, b) system EFP values that are related to the topology (e.g. the worst case execution time) can be calculated using a preformatted and automatic generated Matlab script. The user can use the MathWorks Matlab tool [21] script and the component EFP value data (which can be exported from the PDT) in order to calculate the system EFP values. When all of the system EFP values are calculated (using for instance Matlab or another tool) the user can export the values into a CSV file. |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------- | Table 9.4: PDT - main functionality list |

The tool also provides a Graphical User Interface (GUI). As an example, a screenshot of the GUI is shown in Figure 9.21.
9.7 Appendix: Integrating Evidential Reasoning into MULTIPAR

Based on the work presented in [14], we here describe more in details how ER is applied in MULTIPAR. Specifically, we here present a process flow that details the activities related to ER method. The ER-related process flow is shown in Figure 9.22. The same process can be applied for achieving the component and the system optimizations. With respect to MULTIPAR process flow (see Figure 9.1), we further detailed the prioritization of the weights, the assignment of the preferences and the MCDA ranking. Below the main activities of the ER-related process flow are listed:

- **Decision Matrix Analysis**: the process starts by analyzing the compon-
Figure 9.22: Applying ER method in MULTI PAR.

The decision matrix in order to verify if for each EFP the values of the alternatives are uniformly expressed. For example, if a given criteria is supposed to be expressed by integer value, all the related alternative values have to be expressed by integers. In addition, it will be check if the alternatives values are within the admissible range, e.g. assuming that a property is supposed to have values varying from 100 to 300, all alternative values related to this property will be check against the allowed maximum and minimum values.

- **Weight Assignment and Review**: by selecting a method for elicitation the weights, as for instance [35–38], a weight factor is assigned to all EFPs. The resulting weights are, then, normalized in the range of [0,1].

- **Preference Assignment**: here, a preference is assigned to each EFP in relation to the main objectives. For instance, considering EFPs such as the SDC, the decision makers might desire to have it as lowest as possible, but for the SUP property will be a benefit to have it as highest as possible.
• EFP Hierarchical Modelling and Specification: a hierarchical model of the EFPs has here created. Specifically for ER, this consists of defining the main goal and subsequently sorting and hierarchically arranging all EFPs according to this main goal.

• EFP Subjective Judgement Assignment: here a set of ‘evaluation grades’ to each EFP has to be assigned. For instance, for a qualitative criterion an evaluation set could be defined as WORST, POOR, AVERAGE, GOOD, BEST. For a quantitative criterion the range of possible maximum and minimum values has to be defined.

• Variant - EFP Judgement Mapping: for a given property, the values of the alternatives (both qualitative or quantitative values) have to be mapped with respect to its evaluation grades.

• Variant Ranking: lastly, based on the EFP values and preferences the ranking of the variants under analysis is performed according to the algorithm presented in [18].
Chapter 10

Paper VI: Inclusion of Ethical Aspects in Multi-Criteria Decision Analysis

G. Sapienza. G. Dodig-Crnkovic and I. Crnkovic

Abstract

Decision process is often based on multi-faceted and mutually opposing criteria. In order to provide rigorous techniques for problem structuring and criteria aggregation used for classification and ranking of alternatives, Multiple Criteria Decision Analysis (MCDA) has been used as a method to achieve architectural decisions. Even though it has already been argued in literature that MCDA essentially depends on value systems of decision-makers, it is a question how the decision result reflects a particular criterion, requirement or a particular decision. This is especially true if a criterion is not precisely specified. In this paper we analyse the ethical aspects of MCDA. In our analysis we argue that it is in the long run necessary to make value basis of decision-making and ethical considerations explicit and subject for scrutiny. As a support to encourage introduction of transparent value-based deliberation we propose an extended MCDA scheme that would explicitly take into account ethical analysis. As an illustration, we present an industrial case study for the Software (SW)/Hardware (HW) partitioning of a wind turbine application in which different decisions can be taken, depending on the ethical aspects.
10.1 Introduction

Modern systems are becoming extremely sophisticated and complex, and their development requires taking into account multi-faceted and conflicting preferences and opinions coming from different stakeholders e.g. customers, product managers, project leaders, researchers, system architects, designers, developers, testers and similar. Persons involved in the development influence the decisions taken under the development of these devices in different ways depending on their know-how, experience, personal values, and so forth. As a matter of fact, the choice of the “best decision” always requires the trade-off among different objectives. Thus, regardless of the “measurability of an objective” the relative importance of the objective itself, it is still up to the decision makers, who evaluate it by introducing a certain level of subjectivity. This is because the relative value given to an objective is differently weighted/prioritized from decision maker to decision maker. When considering subjectivity in decision process, the ethical aspects play an important role. We learned from the past errors and we continue to learn from the present how wrong decisions with unjustified ethical perspective have severe consequences for humans and nature today and for the future of next generations. This is valid in general, from the politics and business, to sciences and engineering domain.

There are many different approaches to manage complexity and coming to the decisions based on trade-off analysis. A widespread approach is to apply Multi-Criteria Decision Analysis (MCDA) method. The basic principle of MCDA is to find "a best possible solution" based on different criteria and specified importance (priority) of the criteria. Using different mathematical methods that find an optimal solution of functions that express the weighted criteria, MCDA proposes a solution that does not necessarily satisfy each criterion in the best way, but provides, for given criteria and their weight factors, the best possible "trade-off" solution.

Using results from a MCDA-based procedure, we might come to a solution that fits (considerably) well to the given criteria, but we can ask whether this method omits to analyse the possible ethical aspects and consequences of the decision. Since MCDA does not provide the decision rationale for each criterion, but only the final result, we may ask whether in the decision process the important ethical aspects are forgotten, and the design result is taken for granted. We can ask whether MCDA could also provide a support related to the ethical dimension of the decision(s).

The question we are discussing is this paper is the following:
How can a MCDA-based design methodology be systematically improved so to take ethical aspects in decision making process?

In the paper we analyse MCDA, and propose expanding the MCDA approach presented in [1, 2] with addition to explicitly process ethical aspects related to the given criteria and the decision. In addition we present a concrete case of the embedded systems development where value-based deliberation leads to different outcomes. Our claim is that value basis for decisions must be made transparent in order to be possible to critically assess and harmonize among stakeholders. We focus on a typical issue related to the embedded systems design, which is recognized in the community of the co-design as the partitioning problem [3], [4], i.e. the deployment of an embedded application into HW (e.g. FPGA) and SW (e.g. CPU) computational units.

The rest of the paper is organised as follows. In the Background (Section 10.2) we shortly describe MCDA principles, and continue by pointing out the subjective and essentially value-based side of decision-making process, in spite of the fact that many expect MCDA to present perfect rationality. In Section 10.3 we elucidate the ethical aspects of decision making, addressing cases where neglect of ethical aspects led to severe consequences. In Section 10.4 we focus on the issue of ethical aspects of decision when applying MCDA. We propose a way to augment a MCDA-based design process with ethical deliberation and thus making value-based ethical aspects explicit and transparent. As an example, we present a case study for the development of an industrial wind turbine control systems in which we use MCDA to partition the system into HA and SW components. This is described in Section 10.5. The last section concludes the paper and provides future directions.

10.2 Background

In this section the basic definitions relevant for our research work are presented.

10.2.1 Multiple Criteria Decision Analysis (MCDA).

MCDA is described as “a discipline aimed at supporting decision makers who are faced with making numerous and conflicting evaluations. It aims at highlighting these conflicts and deriving a way to come to a compromise in a transparent process” [5]. MCDA (also referred as Multiple Criteria Decision Aid) is a sub-discipline of operational research and management science, [6] [7] used to “support the subjective evaluation of a finite number of decision al-
ternatives under a finite number of performance criteria, by a single decision maker or by a group” [5]. It is “an umbrella term to describe a collection of formal approaches which seek to take explicit account of multiple criteria” [8] that helps decision-makers to explore decision space. It is widely applied in a variety of fields, such as medicine [9], healthcare, environmental planning, forestry [10], economics and finance [11], energy management [12], transportation [13], public services, marketing, human resources management, and many other fields to support the resolution of decision problems of different nature and complexity [6] [7].

In literature [7], the MCDA problems are often divided into two main classes: the Multiple Attribute Decision Making (MADM) [8] class, which deals with problems that consider a finite number of possible alternative solutions. The second class, referred as Multiple Objective Decision Making (MODM) [14] deals with problem in which there are an infinite number of alternative solutions. A comparison highlighting the main differences between these classes can be found in [15] [14].

In our research, we focus on the application of MCDA/MADM methods to solve the partitioning problem in embedded systems design. In [16] we provided a survey and a suitability analysis of MCDA/MADM methods for partitioning.

The basic elements used in MCDA/MADM are (i) the identified criteria which will be used in the analysis (for example cost, system performance, reliability), (ii) the particular values of the criteria of different solution alternatives (for example price in Euros of the invested efforts, and response time in msec, reliability in percentage), (iii) the prioritization or weighting the criteria (for example price high priority, performance medium priority, reliability high priority), and (iv) the objectives, typically expressed as a function of weighed, or prioritised criteria.

Criteria might be of qualitative or quantitative nature (the price of a house can be numerically measured while the comfort rating can be subjectively described as high, medium, low) or they might be deterministic or probabilistic [17] (the price of a house can be described using deterministic values while the overall impression of location could be a random value). Uncertainty also plays a key role when making decisions. It can be caused by different nature, e.g. as derived by subjective judgement or by unknown or incomplete (imprecise) information [17].

A classification of the most frequent decision-making problems deal with decisions related to the choice, ranking or classification/sorting of alternatives. As a consequence, MCDA methods can be grouped as follows:
• **Ranking methods.** These methods aid decision makers to rank/order the alternatives from best to worst ones.

• **Classification methods.** These methods provide support to the decision makers to classify the alternatives into a predefined classes.

• **Choice methods.** These methods aid the decision makers to identify a subset of alternatives under constraints.

### 10.2.2 Values and subjectivity in the decision process and MCDA.

As widely discussed in the literature, a certain degree of subjectivity is typically involved in the decision making process (see [18] [19]). Inputs to preference models involve subjectivity; weights are values defined by an individual (or a team) and scores also valued from an individual perspective, often based on experience. Values may be measured and precisely specified, but they also can be subjective or a matter of estimation, and it is what MCDA is using as input. As a consequence, using MCDA methods to solve the decision problem implies that among others the developers/designers subjective judgments are affecting the traded-off solution.

Taking into account subjectivity in the decision process, models of rationality must account for judgments being influenced by such factors as experiences and opinions, feelings, beliefs and desires. Kahneman’s [20] dual aspect theory distinguishes between slow (rational, norm-based) and fast (unreflected, emotional) decision making - both of them playing important role. The decision problem is thus projected into a new perspective, a perspective where the solution is also the consequence of subjective value systems, morals, and ethical deliberations, that should be taken into account in a MCDA-based model of decision process. The importance of ethical values and emotions (not captured by the classical model of rationality) when applying MCDA has been already pointed out in [21] [22] [23]. In our study, we focus on values and ethical deliberation that have much wider and socially relevant effects, compared to emotions that are short term and more contingent.

### 10.3 The importance of Ethical Aspects

In this section we introduce ethical analysis tools and outline the existing work, which already established that the value basis of MCDA could be seen through
the lens of ethical analysis, in order to make this deliberation basis visible and open for rational examination.

10.3.1 Consequences of neglecting ethical aspects in decision making process

The technical systems can be designed in different ways, based on the decisions their designers and developers make. Usually it is assumed that the decisions are objective and perfectly rational. However, as already pointed out, perfect rationality is far from engineering in real life, which is never perfect but can be made as good as reasonably possible. We would like to emphasize critical role of shared value systems as a way to minimize negative consequences of subjective components in decision-making process.

Designers and developers are close to the system and best suitable to understand its consequences. Even though individual engineers cannot always influence the development of the whole system, they can make their insights explicit to other stakeholders and thus make better informed decisions.

Recent example of Volkswagen emissions scandal is instructive. It has been found out that cars contained software that would put car under test conditions in a special regime, which produces much less exhaustion. As a result, on the road, the engines emitted up to 40 times more pollutants than was allowed in the US. Reuters news report that “Volkswagen is under huge pressure to get to grips with the biggest business crisis in its 78-year history, which has wiped more than a third off its share price, forced out its long-time CEO and rocked both the auto industry and German establishment”.

What are the factors that might negatively impact the stakeholder’s decisions from an ethical perspective? It might be the inexperience of decision makers, the so called “group think”, which might lead to the tendency to establish entrenched positions or prematurely adopt common perspective excluding contradicting information as well as errors or omissions. A key factor is information communication in the organisation. The importance of taking system-level view of decision making process has been analyzed in another very important application of embedded systems, namely robotics, while addresses ethical aspects of cyber-physical

---


2http://www.reuters.com/article/2015/10/03/us-volkswagen-emissions-idUSKCN0RU15H20151003
systems.

10.3.2 Ethics in technology and other fields with high impact

In the participatory assessment of technology it is important that stakeholders are taken into account. “The grounding assumption is that pluralistic involvement of heterogeneous publics in participatory Technology Assessment (PTA) can assure that decisions are substantively fairer than those that are based upon technical expertise alone” [27]. As “Preferences differ from decision maker to decision maker, so the outcome depends on who is making the decision and what their goals and preferences are” [28]. “The integration of values will result in changes of the MCA understanding, criteria building, and aggregation method, and will not be possible without analytical capacities of the decision analyst in ethics” [29]. As an illustration we can mention the framework programme for European research and technological development, Horizon 2020, that have formulated the Science with and for Society Work Programme, based on Responsible Research and Innovation, where ethical deliberation constitutes the basis.

What does the ethical deliberation imply? Ethics is concerned with study and understanding, systematizing of right vs. wrong conduct. In our case we are interested of applied ethics, which is concerned with a particular application domain, more specifically of software engineering ethics. One of the basic documents describing this field of ethics is Software Engineering Code of Ethics, such as defined by ACM\(^3\) and IEEE\(^4\) codes.

In the process of ethical analysis, which considers stakeholder’s interests and preferences, both intrinsic (i.e. project-specific) and extrinsic (i.e. context-dependent) values are analyzed. We can learn from health technology assessment, where ethics is traditionally integrated MCDA, [30] [31]. Typically, the stakeholder analysis systematization of judgment is made transparent by evidence and values and identification of bias. Here different stages of the process are identified, starting with considering all components of decision, through informing each component of decision consistently, to communicating decisions transparently and supporting understanding and implementation of decisions.

\(^3\)https://www.acm.org/about/se-code
10.3.3 MCDA and Ethics

In [23] Wenstøp analyses the different mindsets of decision-makers starting with the choice of ethics framework. His focus is on virtue ethics [that considers ethical decisions as a consequence of a virtuous character], duty ethics [for which ethical decision is a consequence of obeying duties and norms] and consequentialism [for which consequences of the decision must be anticipated in order to decide if it is ethical or not], which led him to the conclusion that consequentialism and rule-based duty ethics are of importance in practical decision-making, while the virtue ethics acts in an indirect way, through basic attitudes that are making possible ethical deliberation. As value systems in the current setting are largely subjective, Wenstøp argues that “MCDA needs a larger, not smaller, emphasis on values and subjectivity to increase rationality in decision-making”. In commenting Wenstøp’s work, Le Menestrel [21] concludes that MCDA has to be improved in order to capture ethical perspective of the rational behavior. Even Brugha [22] supports this view, making distinction between decision makers needs (basic physical), preferences (cognitive) and values (ethical).

Being subsumed, values stay invisible. Our claim is that in such an essential technology as embedded computer systems, that often is safety critical and mission critical, and always affects us as individuals and society, it is necessary to make value and preferences explicit and subject to critical analysis.

10.4 Explicating ethical aspects in MCDA

The focus of this section is to show how MCDA can be extended to make visible value-based, ethical aspects of decision making. This improvement is supporting the decision makers in considering ethical perspective, and making us aware of the impact that these aspects have in final products. Those deliberations are always made implicitly, and our suggestion is to make them visible in order to critically assess their impact. We address in particular the topic of subjectivity introduced when applying MCDA and during the weight prioritization based on values, where the analysis of system non-functional properties (a.k.a. extra-functional properties - EFPs) is of importance from an ethical perspective. That is a topic that is scarcely elucidated by other authors in our field and it deserves much more attention.

We propose the following steps in explicating ethical aspects of MCDA:

- **Identifying new requirements that are directly related to the ethical is-**
sues. Different projects and their results may have explicit requirements that are directly related to ethical issues. For example exploiting natural resources may directly lead to ethical issues related to the rights and well-being of the local community, for instance building wind mills close to settlements may cause constant disturbance by noise.

• Ethical analysis of requirements and project constraints. Analysis of the application requirements and project constraints from an ethical perspective, which requires all stakeholders involved in the decision process to explicitly address ethical aspects related to the product under development, which will be soon launched to markets. Responsibility for ethics-aware and related issues such as sustainable development is necessary starting from an early stage of the development.

• Ethical analysis of system properties. Identification (if not already included) or analysis of system EFPs that are related, or have impact to the final product from an ethical point of view. Decision makers and stakeholders have to be involved in this activity in order to avoid neglecting ethical aspects. The main objective of this activity is to aid the decision makers in the elicitation of all of the EFPs-related to ethical concerns.

• Ethical concerns in prioritization. Support from an ethical perspective in carrying out the prioritization of the EFPs before the optimization. Weight prioritization plays a key role in MCDA and specifically in the final deployment configuration, as it can be noticed in the proposed case study. Several methods to carry out the weight assignments exist as for instance in [32] [33], but they do not provide any support from an ethical perspective.

• Overall analysis of the system that is a result of MCDA. MCDA defines criteria that are important for particular decisions. These decisions can however have consequences on properties that are not subject of the MCDA. For this reasons (in a similar way as it is done for safety analysis) an ethical analysis should be provided. An example of such analysis is the overall sustainability analysis [34].

At this stage the proposed steps are based on reasoning about ethical aspects following a similar approach is made to extra-functional properties (such as safety and security), in [35,36].
10.5 Case study: HW/SW partitioning for a wind turbine

10.5.1 The partitioning problem.

HW/SW partitioning is considered to be one of the difficult problems when designing embedded applications, as it determines the function, performance, and costs of the product [37]. At the development time, the decisions about an embedded systems partition between HW and SW are of key importance since they impact the product performance and quality, the development process itself, and the product lifecycle up to its disposal [1]. The architects and developers of embedded systems are making decisions that consider many, often conflicting, concerns derived from the runtime and lifecycle requirements and constraints from the project constraints, and business strategic decisions [38].

In our previous work [1] [38] [2] we proposed a novel approach, called
MULTIPAR that is based on MCDA methods, and that supports the developers and managers in taking the decisions to perform the HW/SW partitioning of an embedded application. Key features of MULTIPAR are a) supporting designers in capturing the application requirements and project/business related constraints and transforms them into extra-functional properties (EFPs) (i.e. non-functional properties) where the EFPs represent the decision criteria; b) enabling the reuse of existing computational units; and c) applying MCDA/MADM techniques in order to facilitate a participative decision making process across different decision makers and stakeholders.

10.5.2 Ethical aspects of HW/SW partitioning

Thinking about the impact of software engineering of embedded systems on individuals and society, typically we would not think of the issue of HW/SW partitioning. As it is our field of research interest, we would like to present an analysis of this domain, based on a concrete example of application in order to
show how decision making in different stages is value-based ethical deliberation, and how this aspect can be made visible in the process of MCDA.

Launching in the markets embedded systems products, with the development towards cyber-physical systems, one of the basic requirements is the consistency with sustainable development that is, “the development that meets the needs of the present without compromising the ability of future generations to meet their own needs” [39]. It requires that we involve stakeholders in the development process, with the responsibility of creating products which are environmentally, technologically and socially sustainable.

As example of an ethical reasoning we demonstrate reasoning about sustainability in the design process using MCDA.

### 10.5.3 Case study: HW/SW partitioning

In order to show the impact of the decision makers in the partitioning of an embedded application into HW and SW units, we present an industrial case study. This is focused on the realization of a wind turbine application (WTA). The main objective of a WTA is to control the conversion of the mechanical energy into electrical energy, which will be distributed via a power network. The case study has been developed within the context of iFEST\(^5\), an Artemis JU project. We deployed the WTA into a platform combining a dual core CPU and an FPGA technologies (from the Xilinx Zynq 7000 family [40]). To develop it, we used the MathWorks tools [41] for the design, the automatic code generation and the verification. We applied the MULTI\textsc{PAR} approach [2], and to aid the project manager and developers through the decision process we used the SANNA tool for multiple criteria evaluation of alternatives [42].

The WTA was modelled as a number of interconnected components, an high-level overview of the WTA model is provided in [1]. This latter consists of 7 main components. Each component has associated at least an HW and a SW variant, i.e. component realization. Each variant has associated a number of EFPs [38].

In Figure 10.3 a simplified overview of the decision matrix (DM) shows the components and related variants.

The variants represent the set of alternatives, which are the objects of our study and analysis while the EFPs associated to the variants represent the criteria (in Figure 10.3 only the most relevant EFPs are shown) In order to highlight how ethical aspects can be considered into the design of an embedded system, we illustrate two simple but effective scenarios: the Business-driven scenario

---

\(^5\)iFEST: industrial Framework for Embedded Systems Tools (http://www.artemis-ifest.eu/)
### Table: Decision Criteria

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Component Variant_ID</th>
<th>Decision Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Cr_1</td>
</tr>
<tr>
<td>Sensor Interface</td>
<td>C1.1 SW</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>C1.2 HW</td>
<td>1.7</td>
</tr>
<tr>
<td></td>
<td>C1.3 HW</td>
<td>4.2</td>
</tr>
<tr>
<td>Filter</td>
<td>C2.1 SW</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>C2.2 HW</td>
<td>2.2</td>
</tr>
<tr>
<td></td>
<td>C2.3 HW</td>
<td>1.44</td>
</tr>
<tr>
<td>Main Controller</td>
<td>C3.1 SW</td>
<td>11000</td>
</tr>
<tr>
<td></td>
<td>C3.2 SW</td>
<td>16200</td>
</tr>
<tr>
<td></td>
<td>C4.1 SW</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>C4.2 SW</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>C4.3 SW</td>
<td>17.9</td>
</tr>
<tr>
<td>Required Pitch Estimator</td>
<td>C4.4 HW</td>
<td>1.75</td>
</tr>
<tr>
<td>Pitch Regulator</td>
<td>C5.1 HW</td>
<td>3.5</td>
</tr>
<tr>
<td>Park and Brake Controller</td>
<td>C6.1 HW</td>
<td>3.9</td>
</tr>
<tr>
<td>Supervision System</td>
<td>C7.1 HW</td>
<td>2900</td>
</tr>
<tr>
<td></td>
<td>C7.2 SW</td>
<td>8000</td>
</tr>
</tbody>
</table>

| Business-driven Scenario Weights | 0.3 | 0.6 | 0.1 |
| Sustainability-driven Scenario Weights | 0.3 | 0.3 | 0.4 |

Figure 10.3: Decision matrix. Decision criteria and weights per scenario (Business-driven and Sustainability-driven), and ranking per scenario (see Selected Variant columns).

and the **Sustainability-driven scenario** which are driven by two different sets of value-based decision criteria. Assuming to give the same importance to the performance criterion in both scenarios, the first scenario aims to satisfy the need of launching into the market a new product as soon as possible. Consequently in this case, an higher value (in form of the weight assignment) is given by the decision makers (i.e. the project managers) to the Development Effort criterion, as a project constraint. Diversely, in the second scenario, the interest is to launch in the market a sustainable product from the perspective of lower energy consumption (see the Sustainability criterion). As reported in [43], the power/energy consumption is included in the “green metrics” from a sustainability point of view [44].
The aim of our analysis is to show that value-based, ethically informed decisions have major impact on the partitioning. Different prioritization of the decision criteria (i.e. the weight values) in the development process leads to a completely different HW/SW deployment solution. As it can be seen in Figure 10.3, the main difference is in the Required Pitch Estimator and Supervision System: In the business-driven scenario, the C1.2, C2.2, C3.1, C4.3, C5.2, C6.2 and C7.2 variant were ranked as first alternatives (indicated in Figure 10.3 through the “x” symbol). In the sustainability-driven scenario, the C1.2, C2.2, C3.1, C4.4, C5.2, C6.2 and C7.1 variant were ranked as first alternatives. As it can be noticed that, in the Business-driven scenario with respect to the Sustainability-driven scenario, 5 out of 7 components are deployed as SW units, which obfuscates the fact they lead to an higher power consumption. Conversely in the Sustainability-driven scenario, 5 out of 7 components are deployed as HW units.

This example demonstrates that different reasoning may lead to a different architectural decision. While the presented example addresses primarily problem of energy consumption that is related to a sustainability problem, there are number of other criteria directly related to ethical issues - for example the partitioning decision may have important impact on the system maintainability, but the maintainability might not have been considered as a specific criterion in MCDA, and consequently the customers may remain unaware of this.

10.6 Conclusions and Future Work

Technology is getting increasingly integrated in our daily life, at the same time as it is increasing in complexity. That brings new requirements for functionalities and design in different phases - from initiation and requirements to development, testing and verification and maintenance. Any decision-making inevitably includes subjective elements in which values and ethical norms are part of the decision process. It necessitates involving different stakeholders including customers, designers, developers, team leaders, and administrators, thus representing the whole socio-technological system. In that way values and ethical deliberation become part of quality of the product, with the important new insight that attention should be paid to the relationships with other stakeholders and their value systems and preferences. We propose that value-based ethical aspects, which today are implicit, should be made visible in the course of design and development of technical systems, and thus a subject of scrutiny.

The goal of our present work is primarily to point out the necessity of expli-
cation of ethical basis for values used in MCDA. Thus our argument is on the level of decision making process, shown in Figure 10.2. When it comes to the specific guidance regarding what might be considered into the ethical analysis, one might start from the existing checklist [45] which should be extended and improved through the process of use and learning from experience, taking into consideration stakeholders’ views.

As [46] points out, ethical deliberation in the decision making can be seen as a part of QA process, as “Taking ethical questions into consideration becomes part of the process required for the development of quality software product”. Going further in that direction, we aim in our future work to present a concrete procedure that can be applied for ethical assessment in the development of embedded system.

This study is uncovering tip of the iceberg, and that there is huge complex of issues to explore in the future in order to get systematic and well grounded scheme supporting ethics-aware decision making on different levels of organisation, from individual software engineering practitioners, to the teams and internal organisational decision-makers to external stakeholders. As [47] aptly remark, we are in the stage of the development where “Software engineering evolved from a rigid process to a dynamic interplay of people (e.g. developer and other stakeholders). Organizational and social literature call this interplay an Organizational Social Structure (OSS)”. Their work presents valuable contribution to understanding of the role and forms of organisational social structures in software engineering, and how those can be used in supporting decisions such as choosing the best development scenarios. Research in [47] points towards the necessity of further exploring the role of social networks in software development. In our case, such networks provide mechanisms for negotiations between different stakeholders and decision makers on different levels of organisation and in different stages of the project. OSS study by [47] is aimed at prompt identification of “socio-technical incongruences” as well as learning from experience by detection of OSS that led to failure or success. Here organisational decision-making is of central importance, as Robillard [48] emphasises, which in our case would mean not only learning to use new ethics-aware MCDA tools, but also building an organisational social structure-aware software engineering, acquiring understanding on how the application is to be implemented to avoid social incongruences.

In sum, we see our proposal of ethics-aware MCDA as a contribution to the social software engineering that requires engagement “from a range of stakeholders and end-users working in partnership with multidisciplinary software development teams often at a distance” [49]. This presents new challenges and
opportunities to software engineering, as it presupposes both an agile approach in the software design and development, and the establishment of creative, participatory design processes. We hope in our future work to contribute to the development of MCDA through addition of ethical aspects based on better understanding of organizational and social structures and relationships between involved stakeholders.

The current article is based on reasoning analogous with processes for creating system properties such as safety and security. The next step should be based on evidence, i.e. analysis of existing practice in research. To our knowledge there is no empirical evidence (like systematic literature review) about ethical aspects as requirements or criteria in software development. Furthermore the question of the relation between multiple criteria and decisions, their ethical aspects and the stakeholders should be addressed in empirical studies. Which type of criteria and which decisions have ethical implications and how are they dealt with? The answer to this question will be different in different domains and for different types of projects. Another question of interest is whether including ethical aspects in the requirements and decisions will bring in new stakeholders. Also, an important question is how the stakeholders interact with the decision makers, as the ethical issues often are of larger scope than the concerns of users or customers. These questions could be addressed in case studies.

Acknowledgment

One of the authors (GS) wants to acknowledge the support of KKS foundation through ITS-EASY, an Industrial Research School in Embedded Software and Systems and ORION project, and of the Swedish Foundation for Strategic Research through the RALF3 project.

References


References


References


