Information and Communication Technologies (ICT) Programme

Project N°: H2020-ICT-2016-1-732105

D3.4: CERBERO Modelling of KPI (Ver. 1)

Lead Beneficiary: USI
Workpackage: WP3
Date: 30 January 2018
Distribution - Confidentiality: [Public/Confidential]

Abstract:
[short description of the document]

Document Authors

The following list of authors reflects the major contribution to the writing of the document.

Name(s)	**Organization Acronym**
Christian Pilato	USI
Francesco Regazzoni	USI

Document Revision History

Date	**Ver.**	**Contributor (Beneficiary)**	**Summary of main changes**
30/01/2018	0.1	Christian Pilato (USI)	Initial ToC
31/01/2018	0.2	Francesco Regazzoni (USI)	Added KPI List
15/03/2018	0.3	Francesco Regazzoni (USI)	Wrote the fist version of this Deliverable
13/04/2018	0.4	Francesco Regazzoni (USI)	Modified definitions for including the feedback from Francesca Palumbo, Michael Masin and Julio Filho
25/04/2018	0.5	Francesco Regazzoni (USI)	Added the definitions of KPIs from use case providers
26/04/2018	0.6	Francesco Regazzoni (USI)	Included the inputs from Julio Filho
28/04/2018	0.7	Francesco Regazzoni (USI)	Changed the layout of the presentation, added the design flow
5/05/2018	1.0	Francesco Regazzoni (USI)	Version ready for review
8/05/2018	1.2	Francesco Regazzoni (USI)	Inserted comments from Michael Masin
22/05/2018	1.3	Francesco Regazzoni (USI)	Inserted comments from Karol
24/05/2018	1.4	Francesco Regazzoni (USI)	Inserted comments from Francesca, Evgeny, Michael, and Julio
7/06/2018	1.5	Francesco Regazzoni (USI)	Added references and proposed to close the editing

Table of contents

1. Executive Summary

This document introduces Key Performance Indicators (KPIs) and their definition, and their usage. Contrary to system requirements, which are static and thus suitable only for the design phase of the cyber-physical system (CPS), KPIs are dynamic variables, that express the performance of the system during the whole lifecycle, and that quantifies the discrepancy between the system goal and its current state. KPIs have been successfully used at design time, yet a complete formalization of their use, especially in the CPS world, is still to be achieved. Furthermore, one of the main goal of the CERBERO project is to enable self-adaptation for CPSs. KPIs, intended as dynamic and evolving variables, are the most natural trigger for driving the adaptation process.

The objective of this deliverable is to introduce several fundamental concepts related with KPIs: what they are, how they are defined, and how should be used during the design and the life of a CPS. A crucial step for enabling the design of self-adaptive CPSs is the definition and the incremental population of a library of reusable KPIs. This library will potentially grow during the CERBERO project, according to the use case needs or to the definition of new sources of adaptation for the systems under test. There are several possible families of KPIs We will introduce the ones relevant to the CERBERO project and we will discuss how they are mapped to the models of computation discussed in the Deliverable 3.5 (demonstrating it using a synthetic example).

Finally, we will define the KPIs which will be used in our cases of study. In this phase of the project, we focus on 12 KPIs in total. In this deliverable, we report their generic properties, how they are defined and instantiated in the three use cases of this project, and the way in which they will be measured and evaluated,

1.1. Structure of Document

The document is organized as follows. Section 4 introduces KPIs and the way in which they are modeled. Section 2 introduces the concept of KPIs. Section 3 present our KPI based design methodology. explaining the concept of a library of reusable KPI, the advantages of such an approach, and an example of KPI definition. Section 4 presents several KPIs which are very common for CPSs. Section 5 discuss how the KPIs are mapped to computational models. Section 6 discuss the KPIs which will drive the use cases of the CERBERO project.

1.2. Related Documents

- D2.4 - CERBERO Scenario Description: KPIs are defined on the specific use case described in the updated scenario description (last version 2.4)
- D2.7 - CERBERO Technical Requirements: D3.4 contributes to satisfy D2.7 requirements (KPIs will be used at the design time and during the whole lifetime of the CPSs)
- D3.5 – Models of Computation:: KPIs will be used to represent the system properties which will be verified with different Models of Computation

- D3.6 - Cross-layer Modelling Methodology for CPS: KPIs have to be robust and consistent across different layers and modeling methodologies. This is a fundamental contribution of the CERBERO project.
- D5.6 - CERBERO Framework Components: ultimately, KPIs will be integrated and used by the tools composing the CERBERO framework.

1.3. Related CERBERO Requirements

- Deliverable D2.7 of the CERBERO project [CERBERO_D2.7] defines the CERBERO Technical Requirements (CTRs) of the project (identified with an ID ranging from 0001 to 0020). Research topics related to KPIs activities are reported in Table 1.

- **Table 1: Links to CERBERO technical requirements**

CTR id	CTR Description	Link with the D3.4 document on Modelling KPIs
0001	CERBERO framework SHOULD increase the level of abstraction at least by one for HW/SW co-design and for System Level Design.	Selected Key Performance Indicators are be robust against change of layers of abstraction
0002	CERBERO framework SHOULD provide interoperability between cross-layer tools and semantics at the same level of abstraction.	Key Performance Indicators are defined using a formalism which guarantee interoperability between tools and layers
0004	CERBERO framework SHOULD provide software and system in-the-loop simulation capabilities for HW/SW co-design and System Level Design.	Model-based design space exploration will be supported and driven by the Key Performance Indicators discussed in this deliverable.
0005	CERBERO framework SHOULD provide multi-viewpoint multi-objective correct-by-construction high-level architecture	Key performance indicators allow to define the objectives of the multi-objective architecture.
0007	CERBERO framework SHALL define methodology and SHOULD provide library of reusable functional and non-functional KPIs.	Key Performance Indicators are organized in family of reusable components. Operations, properties and relations discussed for one family are automatically valid for the whole family.
0009	CERBERO SHALL develop integration methodology and framework.	The library of reusable Key performance indicators is a fundamental component of the CERBERO framework.
0020	CERBERO framework SHALL provide methodology and tools for development of adaptive applications.	Key performance Indicators are the way for evaluating the state of the system and to trigger the adaptation.

2. Key Performance Indicators Definition

Key Performance Indicators (KPIs) are a set of quantitative parameters which allows to measure the overall performance of an entity. The term KPI is mostly used in economy, where the entity is usually an organization. Several definitions of what a KPI is have been proposed in the past. Each of them agree on the fact that a KPI should clearly identify the goals of the entity *in the long run* and should allow to measure, in a quantifiable way, the discrepancy of the current performance of the entity from the goals ([ROUB13] [ADEL09]).

In the context of the CERBERO project, the entity is a CPS. However, the general definition of KPI does not change significantly. We can define a KPI in our context, adapting from the definitions coming from economy. Our definition of KPI is as follows:

"A KPI is a quantifiable parameter associated with a metric. A KPI evaluates one critical parameters of a CPS and evaluate the discrepancies from its long term goal".

For CPS, possible examples of KPIs can be the total power consumption, the throughput, the system availability, the system reliability, or the system security. Within the CERBERO framework, to be used for designing CPS and to guarantee their adaptability, KPIs should have the following properties:

- KPIs will always be defined together with a metric which allow to measure them. KPIs has to be quantifiable. This is a crucial step for the successful use of KPIs, since it has to be possible to evaluate the KPI in a clean and not ambiguous way. In most of the cases, the metric depends on the specific system or model of the system which has to be evaluated. The use of generic KPI, as demonstrated in economics, does not lead to meaningful results. Similarly, use KPIs not specifically tailored to the specific system would not be optimal, and ultimately would lead to poor results.

- KPIs are measuring a specific CPS, in the same way that KPIs in economy measure the performance of a specific organization. This means that, to a large extent, the set of used KPIs must be very specific of the target CPS. This is apparently in contrast with the CERBERO idea of creating a library of reusable components. However, despite necessary being ad hoc defined, KPIs will always belong to a specific family. The properties associated to the families, the way a specific family is used in the design process and how they can help simplifying the design and adaptation steps are the reusable elements of the CERBERO library. Possible examples of families are additive families (for instance power consumption is the sum of the power consumption of each actor), ranked families (for instance the preferences of the drivers which prefers highways instead of local road). Each family has its specific properties, expressed with an algebra, which would simplify all the design steps, allow

automating parts of the evaluation and would enable the analytically understanding the properties of a KPI.

- KPI will always be defined with a structured and common formalism. The use of a common formalism has important implications for the KPI based design process. Introducing formalism in the KPI definitions will allow to exploit the full potential of such a design approach, since a formally defined KPI will automatically inherit all the properties and defined for the families to which it belongs.

- KPI definitions could be reused in designing different CPS or provide basis / building blocks for system specific KPIs. In fact, as in the case of object oriented programming, we envision to have a template like definition of each KPI, stored in the library of reusable components. The definition of specific KPIs could be then instantiated in slightly different ways depending on the specific system. For instance, the KPI "area" defined as the sum of the areas of the basic components of the target platform, could be reused to design several CPSs (although sometimes the basic component could be LUTs and FFs, as in the case of FPGAs, or could be the basic gates, as in the case of ASIC).

- KPIs would drive the evaluation of the system during the *whole* live cycle of the CPS, starting from its conception. This means that KPIs, which will allow to evaluate the systems once completed, should be valid also to evaluate the model of the systems, which will be used at design time, in different levels of abstraction.

- In CERBERO, KPIs will also have an additional, very crucial, role. KPIs will be also used to drive the continuous adaptation, exactly as in economy KPIs are used to continuously examine the performance of an entity and to drive the decision process. Because of this, the selection of the specific KPIs and the way in which they are modeled and measures measured are an extremely crucial step since the monitoring and the evaluation has to be integrated with the systems and the actions to be taken will also has to be taken autonomously by the system. An excessive complex model for computing the KPI, although maybe extremely precise, would negatively affect the performance of the system. On the opposite, an extremely efficient, but too approximated, KPI could not allow to reach the level of precision needed by the target application.

Typically, the KPIs lifecycle includes 4 steps [RIOO09]:

- definition: in this step the KPI and a metric to evaluated it are defined and formalized. This step is carried out only once, in our case at the beginning of the design phase.

- measuring: in this step, the metric is applied to measure the KPI. The measuring phase is repeated several times. In our case, the amount of times

that the KPI is measured is typically defined during the design phase, but can be changed during the lifetime of the CPS.

- analysis: in this step, the results of the measuring phase are analyzed and prepared for the decision makers. In our case, the analysis is mostly carried out automatically. Either by a design tool, or by a dedicated engine in the CPS which will continuously compare with defined threshold or drive the optimization process.

- reporting phase: in this step the results of the analysis of the KPI is reported to the decision makers. In our case, the decision makers are mostly tools or runtime managers which will then take the appropriated decision and drive the adaptation.

In this deliverable we will focus on the first two steps, definition and measuring.

Regarding the definition, we will present in Section 3 the main families of KPIs which will be used in the context of the CERBERO project. The list reported in this deliverable will not be exhaustive and will be continuously updated during the overall duration of the project. For each of the identified KPIs, we will provide a general definition and a general modeling methodology and, when needed, we will provide also a use case specific definition.

Regarding the measuring phase, we will define in Section 4 a list of classical KPIs suitable for CPS and the metric to measure them. The metric is part of the KPI definition, and it is needed to allow to clearly and simply assert the performance of the KPI and to evaluate the discrepancy between the goals and the current status of the system. Also in this case, some metrics are very intuitive and sufficiently generic, while some others, where it is very hard to identify a metric, will be inserted as a set of ordered options-, enabling thus a comparison. Finally, as for KPIs, also metrics could be defined in a general way and, subsequently, instantiated on the specific case of study.

3. General Metodology

In this section we discuss the generic approach for classifying, modeling and measuring KPIs, which will be used in the CERBERO project.

The first part of this chapter discusses the creation of a library of reusable KPIs. This library is a fundamental step towards a more systematical and more effective way to design and implement CPSs, as well as a very important contribution of the CERBERO project. The creation, the update, and the maintenance of the library is a task which will continue for the whole duration of the project (from M5 to M28, contributing to Milestones MS2 and MS5). In this chapter we will introduce the concept of family of KPIs, a key concept for the library definition. Then we will discuss the general idea of the library, how it is organized, and how the elements which populates are defined (the way of including the elements in the library will be used also for elements which will be added to the library during the development of the project).

Secondly, we will discuss the way to evaluate the KPIs. If for some KPI a metric could be easily introduced (time, for instance, is measured in seconds, and it is intuitive and natural to identify that an action which is completed in 3 seconds is faster than an action which is completed in 5 seconds), for some others the evaluation process is not so immediate (e.g. is it better to select highway or local roads ?). To overcome this problem, for the latter, together with the definition of the KPI, it is necessary to define a ranking among the possible values that a variable can take.

The last part of this chapter discusses which one are the properties which a KPI should guarantee and enable, and which ones are the main advantages of a KPI based design methodology compared with other design methodologies.

3.1. Library of reusable KPI

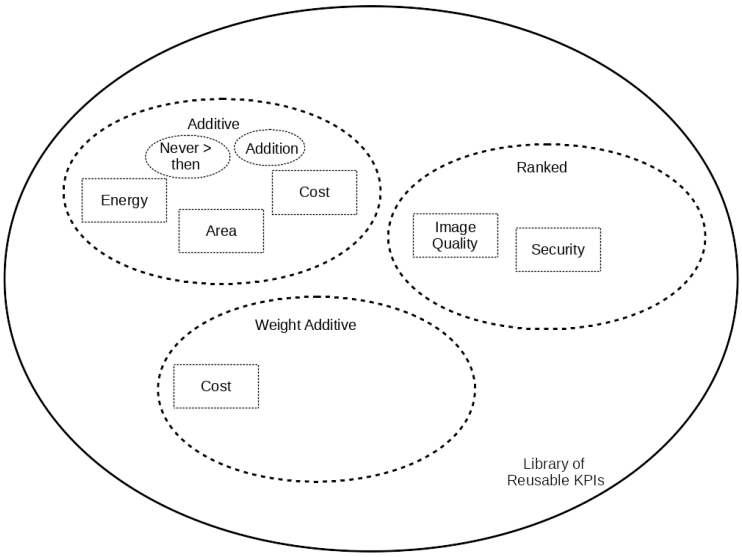
One of the main goals of the CERBERO project is to build a library of reusable KPIs, which will be used as starting point for boosting the design time of CPSs, allowing to address several key challenges of the design phase [LEE08, DERL12],and as key enabler for adaptive CPSs. The population of the library is necessary driven by our use case. In fact, the KPIs which we be included in the library will be the ones which are relevant for our three cases of study, namely Smart Travelling (ST), Planetary Exploration (PE), and Ocean Monitoring (OM). However, the way in which an element is included in the library will be generic, and the abstraction level of several KPI definitions will be sufficiently generic to be suitable for a large number of CPSs (beyond the CERBERO scope).

In the CERBERO project, a KPI is defined together with the way to evaluate it based on the properties of the evaluated system and its elements [MASIN13]. The way to evaluate, will be ultimately expressed with a mathematical structure, or algebra. Properties related with KPIs can defined by the designer or can be inferred by exploiting the mathematical formulation of the KPI itself. The goal of developing a library which is general and reusable is apparently in contrast with the definition of KPIs themselves, which, by nature, are likely to be extremely tailored on the system to be evaluated.

The key concept which we introduce to overcome this limitation is the concept of **family of KPI**. Intuitively, a family of KPIs is a set including all the KPIs accommodated by the same properties and to which the same algebraic operations are valid. More formally, KPIs and the way in which they are evaluated are described with an algebraic formulation. Despite the complexity of its definition, the evaluation of a KPI falls always into some definable *algebra*, and often exhibiting a well know structure. CERBERO uses this fact to formalize the KPIs. In other words, **CERBERO proposes ways to formally define, model, and classify KPIs according to the mathematical structure they exhibit (or need) in their calculations**. We call each class of KPIs obtained in this way family of KPIs.

These properties and these algebraic operations are defined at the family level. Once they are defined for one KPI, they are immediately valid for all the other ones belonging to the same family (also for the one which will be included in the family in future). The properties and the algebraic operations associated to the KPIs are thus the main reusable components.

A simplified version of the families composing the libraries of reusable KPIs are depicted in Figure Illustration, together with some KPIs. The whole library includes KPIs organized as families of KPIs represented in Figure Illustration by the outer ellipse. Each family includes two types of elements: the KPIs (the square boxes of the figure, which contains the definition of the KPI themselves and the definition of the way in which they are calculated) and the properties associated to the family (the dotted ellipses in the figure, which includes the algebraic operations and the properties which the family has and which are valid for each member of the family). In reality, classification of KPI into different families is much more complex, as will be discussed in the remaining part of this section. A KPI can include a hierarchy and be a calculated as a composition of several different families. For instance, in case of several possible path could be taken to complete a task, latency can be defined as the sum of the latency of each single step needed to complete a task for a certain path, and the selection of the path could be defined as a ranked feature by the user.



*SEQ Illustration * ARABIC*

*SEQ Table * ARABICSEQ Table * ARABIC***Figure 1 - Family of KPIs**

For instance, the family additive, includes energy, area and cost. All these KPIs are calculated by an addition operations (e.g., area could be defined as the addition of all the basic blocks composing the design and having attribute "area"). On all the additive KPIs, certain properties are valid and certain operators can be applied. For instance, we can state that the area can never be less than the area occupied by the memory elements included in the design (property *never* < *then* of the figure). Also, we know that we can use the addition operator for reducing the design exploration phase.

As a further step towards reuse of KPI our library will preferably contain the definition of the template of the KPI , which will be then instantiated and customized on each use case, rather than the specific instance of the KPI. For example, the KPI *total area* will be defined as the sum of the basic elements composing the system, which is calculated as a simple sum of the components. This is a template of the definition, since it does not specify which ones are the basic elements (gates, look-up-tables). The properties and the operators available for the KPI total area will be defined also defined over the template. This approach has several advantages. Firstly, it will maximize the possibility of reuse of KPI definitions. Secondly, the template definition will be the same across different levels of the design flow, as it will be detailed in Section 5.

Evaluating KPIs

In this section we discuss how a KPI can be evaluated. The evaluation of any KPI requires some sort of metric. The exact metric is dependent on the target CPS, thus has to be specified by the designer. For some KPIs a metric is intuitive. For some others, it is necessary to introduce a ranking of the possible values which a KPI can assume. Define a metric simply means to determine and formalize a way to evaluate a specific KPI. Such evaluation has to be expressed by means of a certain mathematical structure, which we call an ***algebra*** [MASIN13]**.**

A possible example of KPI is energy. The designer defines energy as the total energy consumption of a system at a given time. A system usually includes many components, each of them, when active, consumes energy. The total energy of the system is computed by summing the energy consumption of each component. Formally, it is defined by the following equation:

$$E_{total}(t) = \sum_{t \in S} E_i\,(t)$$

where *S* is the set of all the components of the system and *t* the time at which the energy consumption has to be evaluated. The set S is updated automatically based on components properties to maintain the value of this equation over the design cycle also when the system is evolving. This is possible thanks to the concept of pluggable algebras, which is extensively used in the CERBERO Intermediate Format (CIF) The formal definition of energy has a specific mathematical structure, the summation over a property of each elements of the set. This allow us to derive some additional

properties for the KPI energy, which are the properties of the summation (such as the commutative and distributive properties). The mathematical structure used, the summation, is one of the KPI family. All the KPI that can be expressed as a summation over a property of each element will belong to this family. All the mathematical properties of the family are immediately valid for all the KPIs belonging to the family.

Another KPI that can be expressed as summation is the total cost of a systems, intended as total monetary cost. Informally, the total cost is the addition of the costs of each block composing the system. More formally, we can define the cost using the following equation:

$$c_{total}(t) = \sum_{t \in S} C_i\,(t)$$

where S is the set of all the components of the system. The mathematical structure of the cost is exactly the same as the one which defines the energy previously defined, but they are applied to a different KPI. In other words, despite they are different, both KPIs are evaluated following the same set of rules and operations, thus they both belong to the family of additive KPIs. Many examples of additive KPIs can be identified in the CPS world (for instance the total weight of a system, the total time of availability, and several others).

Similar to the KPI family exemplified above, it is possible to identify many other families of KPIs which are often used during the design of CPS. In CERBERO we have identified most common families:

- Additive KPI: the KPI is measured by summing over a set of properties of the system
- Multiplicative KPI: the KPI is measured by multiplying over a set of properties of the system.
- Maximum value within a set of properties: the KPI is computed by comparing all the values of a property. The KPI is equal to the maximum value encountered.
- Minimum value within a set of properties: the KPI is computed by comparing all the values of a property. The KPI is equal to the smallest value encountered.
- Average value within a set of properties: the KPI is computed by averaging all the values of a property.

- Hierarchical: the KPI is a composition of one or more of the KPIs indicated above, the composition rule enforces a hierarchy between the KPIs composing the final KPI. Example of hierarchical KPIs are:
    - Weighted sum KPI: the KPI is measured by summing over a set of properties of the system which are weighted using the appropriate coefficient values. The weighted sum can even be over specific components. More in details: the KPI is measured by summing over a specific subset of the properties of the system. In this case, the value of the properties is weighted using the appropriate coefficient value.
    - Weight average KPI: the KPI is measured by averaging over a set of properties of the system over a set of components of the systems.
    - Complex KPI: complex KPIs are defined by a more complex relation between the involved entities. This relation is defined on a use case base. An example is the energy prediction of a system computed over a graph of architectural components. The prediction of the total energy of the system in this case is computed by predicting the energy of each components and then computing the total energy. The energy of each component is multiplicative (instantaneous estimated power consumption multiplied by the time of operation). The energy of the whole system is additive (since is the addition of the energy consumption of each component). Finally, the energy is only a predicted value, which is partially reflecting the real energy consumption (this is because a finer grain computation to achieve precise results would be too complex to be carried out). In this case and in similar ones, the discrepancy between the prediction and the reality should be part of the KPI definition.

KPIs evaluation can imply metrics which are extremely difficult to be defined by means of an analitical function. For these KPIs, where the metric is too complex to be defined and formalized, the KPIs can be defined by the operations and relations needed to calculate it. An example of this is the preference of the highway against local road when selecting the route. It is very hard to formally model the preference of the user using complex analytic functions. These KPIs will be modeled as a list, containing all the possible options, and a partial order between the elements in the list, given by the designer to each of the options. It is important that the introduced partial order is transitive (namely the partial order support the transitivitiy). This way, it would be possible to compare entry A in the list, with entry B in the list and select the best option (for instance, associating to the highway an higher preference value compared to the ones associated to the local road). The necessity of introducing a list with explicit weight is caused by the absence of a practical metrics. An extreme case of this type of KPIs are the ones for which there are only two possible values. For

instance, a designer which needs to encrypt the communication between two components of the CPS using a standard algorithm. The KPI "encryption" in this case, could take only two values, "present" or "not present". Other KPIs will require more parameters, for instance in KPI which are evaluating the throughput of the network, the metric can be calculated only if the network topology is provided. Or, if the KPI values depend on the specific time of measurement, then the designer should also provide, together with the metrics, the dynamic aspects.

Selection of KPIs

The definition and the way in which they are calculated are not the only critical steps to be tackled in KPI based design. A very important phase of any KPI based approach is the selection of the KPIs to be used. The following are the main properties that a KPI should have:

- **be representative of the system**. Selected KPIs will drive the design and the adaptation phase. The selection of KPIs not representative of the system (or the selection of the metric not meaningful for the target system) will lead to a designs which, although reaching the goals defined by the KPIs will not meet the real requirements of the system
- **be robust over design abstraction layers**. Once defined for the target CPS, a KPI should be robust across the different design abstractions and also on the model of the CPS. For instance, if we define energy as the sum of energy of all the components of the CPS, this definition should be valid when we evaluate this KPI at a very high level of abstraction, where the basic component may be an actor, but is also valid at low level of abstraction, where the basic component is a logic gate.
- **be robust against different data sets**. KPIs should also be robust against data sets that can change.

3.2. Advantages of using KPIs

The use of formally defined KPIs during the design phase of the CPS as well as during its whole life cyclebring some advantages and open several possibilities. In particular, the use of formally defined KPIs allows:

- **to enable the automated KPI evaluation** for many KPI families, where the designer just needs to specify part of the data set in the system or the model of the system over which the evaluation should happen.
- **to infer and to early analyze** properties of the systems derived from the KPI families used to evaluate the system, even before the KPI evaluation is carried out. For example, additive KPIs cannot decrease in value when

new components are added to the system (given that there can be no negative property value)

- **to enable complex KPI calculations.** This can be achieved thanks to mathematical formalism such as process algebras. Complex calculation are not limited to only mathematical expressions or equations, but can be even formally described using procedure, complex algorithms or in combinations with other KPIs.
- **to improve DSE.** Exploiting the mathematical properties of families of KPIs, it is possible to improve the performance of DSE and DSE algorithms. For example, some KPI families can be strictly linear, or monotonic, or continuous-by-parts. Exploiting these properties allows to reduce the space to be explored and, ultimately, to improve DSE.
- **to simplify the evaluation of KPIs to support adaptation**. CERBERO targets dynamic and adaptive CPSs. The manager mastering adaptation will be part of the system. The use of strong formalism for specify KPIs which are valid both on the system and on its model, would allow to significantly simplify the adaptation engine.
- **to eases the change of KPIs in adaptive processes**. During system adaptation, KPIs are calculated to decide how the system should change. However, the used KPIs themselves may change accordingly to the evolution of the CPS.
- **to yields to more powerful tools**. Combining the formal models of KPIs with the CERBERO intermediate format would enable a powerful KPI evaluation tool. The *algebras* defined for many usable KPI families map directly into traversal operations over property graphs such as the ones used to implement Cerbero Intermediate Format. This combination further eases automation of the KPI evaluation.

3.3. A complete example of KPI definition

In this section we will discuss a complete example of the definition of a KPI as they are defined in the CERBERO project. We take as example a simple CPS, which continuously senses the temperature of a mechanical components. The temperature must always stay within the operational range (namely it never has to go below or never above specific give values). If the temperature is too high, the system activates the cooling systems, if it is too low, it activates the heating system. However, the heating and cooling system have a cost, so they have to be activated only when is strictly needed.

The system, composed of the needed sensors and actuators, one microprocessor executing a simple routine to compare the temperature value with the reference ones and to issue the decision, and one FPGA implementing the same routines.

Informally, the designer would like to design a system that senses the parameters and quickly decides if the temperature is within the range, or if it is needed to activate the conditioning system in a way or another. The system should operate using minimal energy.

Formally, these requirements can be captured and measured by the following two KPIs: the total energy consumption and the response time to trigger. In details:

- The energy consumption is defined as the sum of all the energy consumption of all the components in the system.

$$E_{total}(t) = \sum_{t \in S} E_i\,(t)$$

- the response time to trigger is defined as the maximum time which passes between the sample of the temperature and the decision of activate (or not activate) the heating or cooling system

$$RT_{total}(T) = Max_{i \in S}\big(RT_i(T)\big)$$

4. Typical KPIs for CPS

In this section we will introduce, as example, several KPIs which are common in CPS. For each KPI, we will provide an intuitive definition, a formal definition, and a metric to evaluate them. We provide a generic definition, which is valid for most CPSs (these definitions are reported also in the glossary of the project), and we define the type of KPI, as discussed in the previous section. The metric provides a way to evaluate the achievement of a specific KPI and it is used to trade off different KPIs. Additionally, in addition to the design phase, the metric will be used during the operation of the system to drive the decision on the adaptivity. Table 1, reported at the end of this chapter, summarizes the introduced KPIs. We will address the use case specific definition in Section 6 of this deliverable.

Latency

Intuitively, latency is defined as "the time need to process a given item". Latency is computed as the addition of the latency of each element involved in the processing of a given item. This KPI belongs to the family of additive KPIs. In reality, in latency can be more complex, for instance in the case of parallel processing. For these situation, latency should be defined on top of processing graph as maximum latency between parallel processing nodes and additive latency between sequential processing nodes. The exact definition is thus use case dependent. Latency is measures measured in the classical unit of time, usually seconds (indicated with s), but depending on specific cases also multiples or fractions of them could be also used. The specific time unit used is use-case dependent and has thus to be defined for each use case.

Throughput

In a nutshell, Throughput is the amount of items processed per unit of time. This KPI belongs to the family minimum, since the throughput of the system is equal to the throughput of the components having minimum throughput (often called bottleneck). In reality, Throughput calculation is more complex. Firstly, it is necessary to compute the delay over the alternative modes. The mode with minimum throughput is then identified as the throughput of the system. The bottleneck can be dependent on the specific used scenario. In this case, it could be needed to compute the average over all the simulation and the scenarios. Also in this case thus, the definition and the calculation of the throughput over a graph could be the best option. Throughput is measured in items per time units, for instance items per second. As in the case of latency, also for the throughput specific use case can use multiples of fractions of the seconds. The exact definition of item is also dependent on the use case and will be defined for each use case.

Energy

Energy is the total amount of energy required by a system to complete a given task. Also in this case, it is necessary to firstly identify the different modes available to complete a task, and then sum up all the energies involved. Depending on the specific scenario, it can be needed to compute the average of the energies of the different cases or just the maximum. The energy consumption is computed as the addition of the energy consumed by each element of the system. This KPI is additive**.** Energy is

measured in Joules (indicated as J). Also in this case, depending on the use case, it is possible to use multiple or fractions. The task to be completed for which energy is computed depends on the used case, thus will be defined in each use case.

Power

Power is the energy transferred per unit of time. Power often is defined as the peak power during the total operation time of the CPS. The power is modeled as the addition of the power consumed by each element at a give time unit. The peak power is identified computing the maximum of the power consumption of the whole system. This KPI belongs to the family maximum, since it is equal to the maximum value of the power consumption. Power is measured in Watts (indicated as W) or in a multiple or fraction of them.

Resources utilization

Resource utilization is defined as the sum of all the resources required by the whole system to complete its operation. Resource utilization is computed as the addition of the resources used by each element. This KPI is additive. Resources utilization are often measured as the total silicon area (measures in mm^2 or in basic components of reconfigurable devices used, such as slices and BRAMs), amount of memory used to store the software code, and maximum amount of memory needed during the operation of the device. These exact definitions are on use case base and target specific.

Quality of service

Quality of service is a measure of the overall performance of the CPS. Quality of service, and as consequence its measure, is an extremely use case dependent KPI (and so is the metric to measures it) and will thus defined for each use case separately. However, in general, quality of service is KPI composed by several other KPIs. An example is a system which is supposed to stay on line. The quality of service in this case can be the amount of time that the system is up and running. In a video streaming which dynamically adapts the resolution to the available resources, the quality of services could be the amount of time that the video is played with an high resolution. In this case, the quality of service is a composition of the running time and a specific threshold in the quality of the video resolution.

Security

Security is probably the most important non-functional requirements for today CPS [ALFA15]. This KPI is use case specific, since the definition of security as requirement is very much depended on the specific scenario and the possible threats which the CPS could face. We generically define security as the composition of the four main properties which secure systems should provide, namely confidentiality, integrity, non repudiation and availability. Each use case will define which one of these four properties have to be included. Security of the CPS is the computed as the logical “AND” between them. If at least one component of the system does not guarantee the required properties, then the system is not considered secure. If the all the components of the systems are providing the required properties, then the system

is considered secure. Security will be measured as a true/false variable according to the fact that a specific required functionality is present or not in the system.

Reliability

Reliability measures the capability of a system to operate for a given amount of time. This KPI belongs to the family minimum , since the reliability of the system equals the one of the least reliable point. In practice, the computation of reliability strictly depends also on the topology of the system. For example if minimal reliability assigned for 2 identical parallel components that can perform same actions, then reliability of the whole system is not minimum of this 2, because if one of this will fail second can perform its functions. Classical way to measure reliability includes the mean time to failure (measure in time units such as seconds) and the availability (which, in this project will also include the mean time to repair the system).

Response time to triggers

Response time to triggers is the time needed by the system to react with an action to a specific trigger (for instance, the time needed to reconfigure the system, once the reconfiguration signal is issued). This KPI belongs to the family minimum, since the response time to trigger of the system equals the one of the least responsive one. Response time to triggers is measured in time units, such as seconds, multiple or fractions.

Availability

Availability is a measure of the availability of the component, intended as answer to the question "is the component available in an existing library?". This KPI is measured as a true or false, depending on whether the component is available or not.

Cost

Cost identify the total cost of a CPS or of a solution proposed by the CPS. This KPI is additive. However, the exact way to measure cost, is strongly use case dependent. As a result, it will be defined on a use case base.

Table Table summarizes the KPIs we introduced in this chapter reporting their basic type and the unit used to measure them. It is important to underline that most of the KPI presented are in reality much more complex to be defined and extremely dependent on the use case.

KPI	**Basic Type (exact type is use case dependent)**	**Unit**
Latency	Hierarchical Additive	Seconds s (multiple or fractions)
Throughput	Hierarchical Minimum	Items per seconds (multiple or fractions)
Energy	Additive	Joule J (multiple or fractions)
Power	Additive	W (multiple or fractions)
Resources utilization	Additive	Silicon area, basic blocks
Quality of service	Hierarchical Minimum	Use case dependent
Security	Minimum	True or false
Reliability	Minimum	Mean time to failure, availability
Response time to triggers	Maximum	Seconds s (multiple or fractions)
Availability	Ranked feature	True or false
Cost	Additive	Use case dependent

Table 4 Summary of typical KPIs of CPS. The table reports the type and the unit of measure.

5. Mapping to computational models

In this chapter we discuss how KPIs can be mapped to computational models. This step is important, since it guarantees the link between the modeling phase of the project and the design and adaptation phase of the project.

Models of computation (MoC) are introduced and explained in Deliverable D3.5. The same deliverable extensively explains the MoCs which will be used during the CERBERO project and the tools which will be used to support them. In this chapter we quickly recall what are the models of computation and we discuss how the KPIs we introduce can be used and mapped to these MoCs.

A Model of Computation (MoC) is a set of operational elements that can be composed to describe the behavior of an application. The set of operational elements of a MoC and the set of relations that can be used to link these elements are called the semantics of a MoC. Each element of the semantics of a MoC can be associated to a set of properties such as timing properties or resource requirements. These rules and properties provide the theoretical framework that can be used to formally analyze the characteristics of applications described with a MoC. In CERBERO part of these rules will be expressed by the KPIs.

For example, using a mathematical analysis, it may be possible to prove that an application described with a given MoC will always guarantee a specific response time to trigger or that the power consumption of the computation will never go above a certain threshold.

Table Table summarizes the tool chains which are used and are developed within the CERBERO project and reports the supported model of computations.

	SDF	PiSDF	PN	KPN	DPN	RTL	DES	SCE	TS
MECA								S	
VT									S
DynAA			S	S			S		
AOW									
PREESM	S	S							
SPIDER		S							
PAPIPY		P			S				
JIT HW						S			
ARTICo^3						S			
MDC	S	P			S	S			

5. Table: Mapping between MoCs and Tools (from Deliverable D3.5). S: supported, P: planned

In the last part of this section we report how these tools may benefit from the use of KPIs and which KPIs will be mapped to them in the following months of the project. We will include also KPIs which are use case specific, and thus defined in Chapter 6. As previously discussed, a tool which is extended to support a KPI belonging to a

specific family, will be immediately capable of supporting all the KPIs belonging to the same family.

MECA: one main KPI will be modeled with MECA:

- user satifaction

DynAA: three main KPIs will be modeled with DynAA:

- response time to trigger (where the trigger can be the response time to provide requested data or to the need of reconfiguration due to a change of the computation environment).
- battery lifetime

These KPIs are belonging to the family additive and ranked feature.

AOW: three KPIs will be modeled using AOW:

- monetary cost of the CPSs
- network communication
- energy transferred or stored by the system
- throughput
- latency

These three KPIs belong to the family additive.

PREESM: four KPIs will be modeled using PREESM:

- energy consumption
- throughput
- resource utilization (in terms of area of basic components)
- reconfiguration time

These KPIs belong to the family additive, minimum and maximum.

SPIDER: three KPIs will be modeled using SPIDER:

- execution time
- latency
- energy consumption

These three KPIs belong to the family additive.

ARTICo^3: four KPIs will be modeled using ARTICo^3:

- energy consumption
- throughput
- resource utilization (in terms of area of basic components)
- reconfiguration time

These KPIs belong to the family additive, minimum and maximum.

MDC: four KPIs will be modeled using MDC:

- energy consumption
- throughput

- resource utilization (in terms of area of basic components)
- reconfiguration time
- QoS

These KPIs belong to the family additive, minimum and maximum and ranked features.

	Additive	Hierarchical	Minimum	Maximum	Ranked Feature
MECA					
VT					
DynAA	X	X			X
AOW	X	X			
PREESM	X	X	X	X	
SPIDER	X	X			
PAPIPY					
JIT HW					
ARTICo^3	X	X	X	X	
MDC					x

6. Table: Mapping between KPI families and Tools which will be demonstrated during the CERBERO project

6. KPIs within use case

In this section we discuss the definitions of KPIs and the metrics that are used to measure them in CERBERO use cases. In this chapter we mainly concentrate on metrics which are not generic, thus the one which needs to be defined on the use case base.

6.1. Smart Traveling

The Smart Traveling use case address the problem of driving assistance for electric vehicle, considering several parameters and constrains, including the insurance of the sufficient level of battery, use driving styles, and different types of cars. In this section we will introduce the KPIs for this case of study.

Latency

Latency within the Smart Traveling use case is the delay time of signals within the system. The latency is for example related to generation of sensor data in the simulator and receival of this data by the server and/or application which needs to process the data. This type of KPI is additive, since the latency of the whole system is equal to the sum of the latency of each element involved in the communication.

Throughput

Throughput within the Smart Traveling use case is the amount of data transmitted from sources (e.g. sensors) to destinations (processing services in for example DynAA or MECA). This KPI belongs to the family minimum, since the throughput of the whole system is equal to the one of the component involved in the transmission having the minimal one.

Energy

Energy within the Smart Traveling use case is the electric energy transferred or stored within car, such as energy stored within the battery or energy used by the motor or electric devices within the car such as lights and heating or cooling. This KPI belongs to the family additive.

Power

Power within the Smart Traveling use case is the (fictive) energy needed to accelerate and drive the car from the origin to the destination, taking into account items like declinations, accelerations, road and weather conditions. This KPI belongs to the family maximum.

Resources utilization

Resources utilization within the Smart Traveling use case is the optimal usage of main resources in the use case: battery storage, utilization of charging poles, time for travel and time to load.

Quality of service

Quality of service is the overall performance of the simulations executed for the of the Smart Travelling scenario and related to:

- Correctness of events and responses to events (indicating by how realistic the events and related responses are);
- The precision of the simulations (indicating the level of detail in which system behavior is simulated);
- The accuracy of the timing of events and responses.

In all the three definitions, this KPI is a ranked feature, since the designer needs to specify additional information to introduce a ranking between different results.

Reliability

Reliability within the Smart Travelling use case is the realistic accuracy of the simulation (how realistic the simulation is compared to a real traveling experience using a real electric vehicle). This KPI is a ranked feature, since the designer needs to specify additional information to introduce a ranking between different results.

Response time to triggers

Response time to triggers in the Smart Travelling use case is the speed (number of seconds or milliseconds) in which the simulation can respond to triggers from either the simulation modules of physical controls (of the car simulator). This KPI belongs to the family minimum.

Availability

Availability within the Smart Travelling use case is the measure of availability of essential services for the travelling trips like charging poles and traffic free roads. This KPI is a ranked feature, since the designer should introduce, together with the metric, additional information regarding the topology of the road, the position of the poles.

Cost

Cost within the Smart Travelling use case are:

- Financial cost in Euros related to for example cost of energy and cost of (fast) charging;
- Psychological costs expressed in compliance with defined user preferences (such as preference for green scenarios while driving or preferences for highways)
- Cost in travel time (total number of seconds or minutes to reach destination)

In the first and the third definition, this KPI is additive. In the second it is a ranked feature, since the designer needs to provide additional information regarding the used preferences to allow the comparison between different options.

Additional KPIs:

- Type of charging (Charging slow, charging fast)
- Occupancy of charging poles (free, reserved, occupied, unknown);
- Road status (empty, quiet, busy , very busy, traffic jam);
- Road type (flat, uphill, downhill).

All these KPIs are ranked features, and the designer has to provide an order between the different preferences to allow to compare the different options.

6.2. Self-Healing for Planetary Exploration

Self healing for planetary exploration explores the use of self-monitoring and self-healing capabilities to overcome the failures caused by cosmic radiations. The status of the system is measured by high performance sensors and the self-healing is guaranteed by the reconfiguration. In this section we define the KPIs needed in this use case.

Latency

It is defined as delay before the robotic arm starts moving when all kinematic equations are solved. Inverse kinematics equations can be solved by using parallel methods of calculation (HW), which reduces the latency of the system. This KPIs is of type additive.

Throughput

Maximum rate that inverse kinematics equations can be solved. Reconfiguration, adaptation and HW accelerators improve the throughput (e.g. reducing calculation time of inverse kinematics equations). This KPI is of type minimum.

Energy

Energy consumption of the system due to equations calculations and movements of the robotic arm. Providing energy measurement and adaptation to optimize energy consumption. This KPI is of type additive.

Power

Power consumption of the system due to equations calculations and movements of the robotic arm. Providing power measurement and adaptation to optimize power consumption. This KPI is of type additive.

Resources utilization:

Resources utilization in rad-tolerant Zynq FPGAs. By HW/SW adaptation, the system is able to use SW (PS) and HW (PL) resources. This KPI is additive.

Security

Confidentiality in communication with the robotic arm. Communication with the robotic arm will be encrypted using standard space algorithms, in order to guarantee the confidentiality of data in critical applications. This KPI is a ranked feature.

Reliability

Reliability of the system in harsh environments with radiation effects. By formal verification and adaptation methods, reliability of the system can be guaranteed. This KPI is a ranked feature.

Response time to triggers

Reconfiguration time due to radiation effects. Different topologies will be used to analyze response time to triggers and reconfiguration. This KPI belongs to the family maximum.

Availability

Availability of communication with the robotic arm. By adaptation and reconfiguration, availability of the system can be guaranteed, avoiding communication loss with the robotic arm due to radiation effects. This KPI is of type ranked feature.

Cost: the cost in this case is the development cost. A toolchain environment for the design of CPS will reduce developing time and costs. This KPI is additive.

6.3. Ocean Monitoring

The Ocean Monitoring use case exploits video-sensing, mounted on underwater ocean robots, to serve as marine eyeballs that can capture live videos and images of the local on-sea and subsea surroundings. These robots may be guided from the shore or operate autonomously. We define hereafter the specific KPIs used to design, operate and evaluate this use case.

Throughput

In the OM use case, throughput is mainly related with the throughput of the communication. It is defined as the amount of data transferred per time unit within a given communication channel (Bluetooth, Ethernet, GSM, WiFi, 4G, etc.). Within the OM use case and context of operating a marine robot, there are several aspects.. e.g., the camera data buses are limited by throughput, as is video compression, writing to storage, and so on. There will be quite a few resource limited connections at various parts internally within the OM case. It is a KPI belonging to the family minimum.

Energy

Energy within the Ocean Monitoring use case is the amount of energy available in a battery used for the motor and electronic equipment in a marine robot. This KPI belongs to the family, since it is equal to the maximum amount of energy present in the battery at a given time frame.

Power

In the Ocean Monitoring use case, it is the power needed for propulsion, steering, and other digital equipment onboard the marine robot such as sensors, camera system, read/write to storage, operation of wireless communication. This KPI is additive, since the power needed for each of the components of the marine robot can be summed up to determine the power needed by the whole system.

Response time [to triggers]

In the OM use case, response time to trigger is related with the time needed to retrieve the data once they are requested. This KPI is of time maximum, since the response time is identified by the worst case response time.

Cost

Cost within the OM use case are:

- Financial cost to prepare robot for trip e.g. cost of charging, cost of preparing parts.
- Time cost to prepare robot for trip and duration of trip.

Both of the definitions of cost used by this KPI are additive.

Image Quality

In OM use case, image quality can be defined in terms of measurable characteristics, perceived degrees of quality, or a combination of these. Measurable characteristics can include thresholding, for example better resolution of an image can be achieved by using the Binarization method of optimum thresholding techniques. Other measurable characteristics include specific types of image degradation, for example blurriness or sharpness, noise level, resolution, dynamic range, contrast, and colour depth. It is also possible to refer to image quality as perceived image degradation in comparison with the reference or perfect image. Other perceived degrees of quality include the notion of a "fitness for purpose", task or situational relevance i.e. where an image may match some observable characteristics but in practice not be useful for the task or purpose at hand. For example, a high quality picture of a fish might need to preserve the information needed to identify the fish. Since overall it is very hard to give a definition of image quality which suites the purpose of the use case, we will consider image quality as a ranked feature, and the ranking between different options in the achieved quality will be given by the use case provider.

7. References

[ADEL09] Adela del-Rio-Ortega, Manuel Resinas; *Towards Modelling and Tracing Key Performance Indicators in Business Processes*, Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos, Vol. 3, 2009

[ALFA15] Al Faruque, Mohammad, Francesco Regazzoni, and Miroslav Pajic. "Design methodologies for securing cyber-physical systems." *Proceedings of the 10th International Conference on Hardware/Software Codesign and System Synthesis*. IEEE Press, 2015.

[DERL12] Derler, Patricia, Edward A. Lee, and Alberto Sangiovanni Vincentelli. "Modeling cyber–physical systems." *Proceedings of the IEEE* 100.1 (2012): 13-28.

[LEE08] Lee, Edward A. "Cyber physical systems: Design challenges." *Object oriented real-time distributed computing (isorc), 2008 11th ieee international symposium on*. IEEE, 2008.

[MASIN13] Michael Masin, Lior Limonad, Aviad Sela, David Boaz, Lev Greenberg, Nir Mashkif, and Ran Rinat. "*Pluggable Analysis Viewpoints for Design Space Exploration"*, CSER 2013, Procedia Computer Science 16(0), 226-235, Elsevier, 2013

[RIOO09] del Rıo-Ortega A, Resinas M, Ruiz-Cortés A. Towards modelling and tracing key performance indicators in business processes. II Taller sobre Procesos de Negocio e Ingenierıa de Servicios, PNIS. 2009.

[ROUB13] Ella Roubtsova, Vaughan Mitchell; *Modelling and Validation of KPIs*, Proceedings of the Third International Symposium on Business Modelling and Software Design, 2013