Res Eng Design (2017) 28:333–356
DOI 10.1007/s00163-016-0243-2

ORIGINAL PAPER

# Multidisciplinary design methodology for mechatronic systems based on interface model

**Chen Zheng[1] · Peter Hehenberger[2] · Julien Le Duigou[1] · Matthieu Bricogne[1] · Benoît Eynard[1]**

Received: 19 August 2015 / Revised: 27 September 2016 / Accepted: 27 October 2016 / Published online: 9 November 2016


**Abstract** Mechatronic system is considered as the resulting integration of electrical/electronic system, mechanical parts and information processing. Therefore, to enable a systematic design process of mechatronic systems with a high-level integration, the so-called multidisciplinary integrated design is required. However, neither academia nor industry has yet provided an effective solution, which can fully support the whole design process to achieve such multidisciplinary integrated design. In order to organise the design activities from different disciplines and to aid the designers to achieve the multidisciplinary integrated design, the authors propose a design methodology based on a multidisciplinary interface model. In line with the systems engineering practices, an extended V-model is used as the macro-level process in the proposed design methodology. It starts with identification of requirements on the entire system and ends with a user-validated system. The hierarchical design model is adopted as the micro-level process. It supports the specific design phases where individual designers can structure design sub-tasks and proceed and react in unforeseen situations. To ensure the consistency and traceability between the two levels, the multidisciplinary interface model is proposed. This design methodology is demonstrated by studying the design process of a quadrotor.

✉ Julien Le Duigou
julien.le-duigou@utc.fr

[1] Department of Mechanical Systems Engineering, Université de Technologie de Compiègne, Sorbonne Universités, CS 60319, 60203 Compiègne Cedex, France

[2] Institute of Mechatronic Design and Production, Johannes Kepler University, Linz, Austria

**Keywords** Mechatronic design · Design methodology · Multidisciplinary integration · Interface modelling

## 1 Introduction

The term mechatronics originated at the Yaskawa Corporation from the combination of mechanics and electronics in 1969. After the 1970s, the meaning of mechatronic has been broadened to include software and computation (Carryer et al. 2011). Nowadays, mechatronics becomes the synergistic integration of mechanical parts with electrical/electronic systems and information processing (Tomizuka 2002). With the development of technology, other disciplines (e.g. optics, hydraulics and pneumatics) are involved in the development of mechatronic systems (Gausemeier et al. 2009).

As mechatronic systems encompass a wide range of disciplines, the design of mechatronic systems requires a multidisciplinary integrated design (Abramovici and Bellalouna 2007). The design methodology can help the engineers from different disciplines to enable their collaboration for increasingly complex tasks (Hazelrigg 1996). However, neither academia nor industry has yet provided an effective design methodology, which can fully support the engineers to achieve such multidisciplinary integration (Zheng et al. 2014; Bricogne et al. 2014).

In order to overcome the limits of current design methodologies for the design of mechatronic systems, the paper proposes a design methodology based on a multidisciplinary interfaces model. This methodology is divided into two levels, called macro-level and micro-level. The two-level structure can be found in previous studies (Gausemeier and Moehringer 2003; Jansen and Welp 2005; Hadas et al. 2010; Brezina et al. 2011). The macro-level

process is used to describe the generic procedure for the design of mechatronic systems, and it can be specified according to the individual design phase which is called the micro-level process. The micro-level process supports every specific design phase where individual designers can structure design sub-tasks and proceed and react in unforeseen situations. The proposed design methodology adopts an extended V-model as the macro-level process and the hierarchical design model as the micro-level process. A multidisciplinary interface model is proposed to ensure the consistency and traceability between the two levels.

The term "interface" used in the paper is defined by the authors as "the logical or physical relationship integrating the system components of one mechatronic system or the components with their environment". The multidisciplinary interface model can provide an effective guidance for the organisation of design activities and support the collaboration of designers during the design process. Therefore, the proposed design methodology based on the multidisciplinary interface model will help the designers to focus and achieve a multidisciplinary integrated design for mechatronic systems.

The paper is organised as follows:

Section 2 reviews the current design methodologies, V-model and its variants, hierarchical design model and interface model of current product models. Their limitations in multidisciplinary integrated design will be pointed out in this section. Section 3 presents the macro-level design process (the extended V-model), the micro-level design process (the hierarchical design model) and the realisation of consistency between the two levels. A case study of the design activities based on a quadrotor is then introduced to demonstrate the proposed design methodology in Sect. 4. Section 5 provides a detailed discussion on the proposed design methodology. Finally, the authors draw the conclusion in Sect. 6.

## 2 Literature review

### 2.1 Current design process models and design methodologies

To achieve a multidisciplinary integrated design for the design of mechatronic systems, one of the most significant issues is to overcome the "Process-based problems" (Abramovici and Bellalouna 2008). The "Process-based problems" are described as "the coordination and synchronisation of discipline-specific development process, the coherences and interactions between the different disciplines and comprehensive integration across all disciplines" (Abramovici and Bellalouna 2007). Various design process models and design methodologies focusing on multidisciplinary integration have been proposed, and they are all considered as a potential solution to such "Process-based problems".

The traditional approach for the design of mechatronic systems is called sequential design process. In this design process, the main concerns of the mechanical view are reliability and technical performance of the system. The control view of the system is then designed and added to provide additional performance or reliability and also to correct undetected errors in the design (Alvarez Cabrera et al. 2010). According to the definition given by Shetty, the sequential design is "a traditional design process of keeping engineering disciplines separate" and the design activities carried by engineers of various disciplines work on a project occur sequentially (Shetty and Kolk 2010). In sequential design process, new discipline-specific design task cannot start before the previous one has been finished. For example, the mechanical design has to be "frozen" before proceeding to the design of control software (Shetty and Kolk 2010).

Above presentation shows the process-based problems related to the integrated design of mechatronic systems contrast the sequential design process since the sequential design process is the inherently complex nature of designing a multidisciplinary system (Shetty and Kolk 2010). Although the sequential design process can help executive managers to have a global view about the whole design, it is not suitable for modern industries any longer because the whole duration of the design process is very long (the design in each discipline has to be carried out one after another) and the sequential design process does not reach a high integration level. The result of a survey carried out by Aberdeen researchers reveals that the companies in which concurrent design process is used prove to be much more likely to hit product cost targets and product launch dates than those who adopt sequential design process (Lock 2009; Shetty and Kolk 2010).

Since the late 1950s and the early 1960s, systems engineering has been proposed as a multidisciplinary approach and means to enable the realisation of a concurrent design process for complex systems (Blanchard 2012). A concurrent design process is a means to decrease the project lead time and to harvest the synergy of a multidisciplinary design (Torry-Smith et al. 2013). In the 1980s, some system design methods, such as waterfall model (Boehm 1981), spiral model (Boehm 1988) and V-model (Forsberg and Mooz 1999), were widely used for systems engineering, but a design method specially adapted for design of mechatronic systems was not put forward at that time. Since the 1980s, the continuously growing complexity of mechatronic systems has required a more integrated design than ever. Therefore, a number of

mechatronic design methods have emerged to meet the need of multidisciplinary collaboration during the design process of mechatronic systems. From approaches such as the traditional sequential design to the concurrent engineering (Li et al. 2001) or the much recent lean product development (Gautam and Singh 2008), many design methods have been proposed, but these design methods still remain poor to support the multidisciplinary perspectives in the design of mechatronic systems (Zheng et al. 2014).

The field of design methodology is a rich collection of findings and understandings, resulting from studies on how the designers carry on the design process. Tomiyama et al. (2009) define the design methodology as "it is about design processes and activities". A non-exhaustive list of current design methodologies is presented hereafter.

Model-based systems engineering (MBSE) has attracted numerous researchers' attention recently. It is considered as a significant methodology for the design of mechatronic systems with the increasing complexity (Fisher 1998). The Object Management Group's System Modelling Language (SysML) has been widely used to support the MBSE recently (Object Management Group 2009). Some extensions have been developed for SysML to support the specific requirements of design of mechatronic systems, such as automatic simulation (Cao et al. 2011), dependency modelling (Qamar and Paredis 2012) and design-making process (Kerzhner and Paredis 2012). However, current studies on MBSE mainly focus on early design phases and seldom involve the detailed design phases. Moreover, a systematic design process supported by a uniform methodology which integrates all the extensions still needs to be further developed.

Function-Behaviour-State (FBS) modelling has been used to analyse the functional structure in the functional design phase for complex systems for a long time (Umeda et al. 1990, 1996; Hamraz et al. 2014). Several extensions of FBS modelling approach and design methodologies for mechatronic systems have been developed by the research team of Tomiyama (Alvarez Cabrera et al. 2010; Komoto and Tomiyama 2011, 2012). However, like the MBSE, the methodologies based on FBS also focus on the early design phase, i.e. descriptions of the decomposition result from functional, parameter-level, structural and behavioural aspects. How to ensure a correct integration after the decomposition process is seldom involved.

Characteristic properties modelling/property-driven development (CPM/PDD) developed by Weber provides new explanations of phenomena that have previously been insufficiently understood in design theory and methodology for complex systems (Weber 2005). CPM/PDD does not claim to be a new or alternative design methodology; it provides a framework that can be integrated many existing approaches. However, CPM/PDD remains in the area of mechanical engineering, and electrical/electronic and software components are not well covered (Weber 2014).

By analysing the literature review, the main current design process models and design methodologies and their limitations are summarised in Table 1:

In order to break through the barriers related to design process models and design methodologies, the paper proposes a design methodology combining the macro- and micro-level processes. For the macro-level process, an extended V-model is used. In next section, the current V-model and its variants are reviewed.

### 2.2 V-model and its variants

As depicted previously, systems engineering has been proposed as a multidisciplinary approach to enable the realisation of complex systems. The V-model is considered as an effective way of representing the systems engineering and development process. It starts with identification of requirements on the total system. Once the requirements have been taken into account, they are then placed under project control (upper-left) and the V-model will end with a user-validated system (upper right). In order to achieve the final product, each stage of the product definition should be tested (Forsberg and Mooz 1999).

The V-model defines a multidisciplinary design process for the complex system, but it remains generic to support the design of mechatronic systems. During the phase of implementation, the V-model simply divides a complex system into software and hardware, ignoring the fact that the mechatronic system is the combination of mechanics, electronics and software. For that reason the multidisciplinary integrated design is partially performed in V-model.

Some design methods using the V-model as the macro-level have been specially put forward for the design of mechatronic systems. The macro-level of design process means that the generic process is related to design phases and corresponding product states (Gausemeier and Moehringer 2003). The VDI guideline 2206 (VDI 2206 2003) is developed and standardised by VDI committee, a German engineers association. It represents a practice-oriented guideline for the systematic development of mechatronic systems based on the V-model (Fotso et al. 2012). Compared with the V-model, the VDI guideline 2206 unifies the discipline-specific design more systematically (mechanical engineering, electrical engineering and information technology), but the specific design activities in every design phase have not been specified (Fig. 1).

Some design methods that possess several passes of V-models as macro-level have been proposed because a single V-model is understood as a generic procedure pattern and a complex mechatronic product will normally not

**Table 1** Summary of current design process models and design methodologies and their limitations

| Design process models and design methodologies | Limitations |
|---|---|
| Sequential design process (Shetty and Kolk 2010) | Does not allow a concurrent engineering |
| Waterfall model (Boehm 1981); Spiral model (Boehm 1988); V-model (Forsberg and Mooz 1999); CPM/PDD methodology (Weber 2014); Concurrent engineering (Li et al. 2001) Lean product development (Gautam and Singh 2008) | Is not specially adapted for design of mechatronic systems |
| MBSE and its extensions (Fisher 1998; Cao et al. 2011; Kerzhner and Paredis 2012; Qamar and Paredis 2012) | Only focuses on the conceptual design phase |
| FBS and its extensions (Umeda et al. 1996; Alvarez Cabrera et al. 2010; Komoto and Tomiyama 2011, 2012) | |

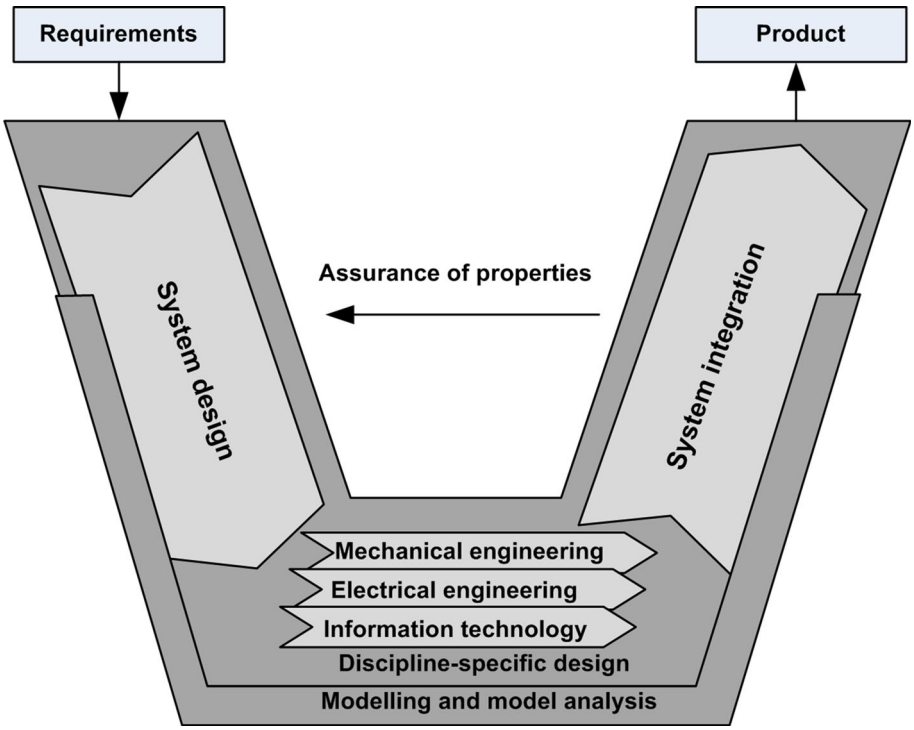



**Fig. 1** V-model in the VDI guideline 2206 (VDI 2206 2003)

be finished within one macro-cycle (Gausemeier and Moehringer 2003). Vasić and Lazarević (2008) believe that the product maturity (i.e. "laboratory specimen", "functional specimen" and "pilot-run product") should be taken into consideration during the design process; thus, several V-models should be adopted in order to represent the product maturity. Hofmann et al. (2010) also insist that a number of macro-levels should be required for the design of complex mechatronic systems. They propose an additional V-model to represent the reliability information flow during the design process. Gausemeier et al. (2011) propose a 3-cycle model. In this model, two V-models are used to represent the virtual product development and the virtual production process, respectively. A W-model based on the V-model is proposed for the development of mechatronic systems. Two V-models are linked together to represent five design phases: "System analyzing", "Specific solutions and dependency analysis", "Virtual system integration", "Model analysis and detailed development" and "System integration". Central element is "Virtual system integration" defining the name giving "W"-shape (Nattermann and Anderl 2013; Barbieri et al. 2014). However, all the design methods possess several passes of V-models as the macro-level process, but none of

them support the specific design phases in which individual designers can structure design sub-tasks and proceed and react in unforeseen situations, i.e. micro-level process.

Attentions have been paid to the micro-level process by recent studies, and some variants of V-models have been proposed. Bathelt et al. (2005) applied the extensions for every design phase based on the VDI guideline 2206 in order to improve the development of systems controlled by a programmable logic controller (PLC). However, due to the speciality of systems controlled by PLC, the discipline-specific design phase is simply divided into 3D CAD part and PLC programming environment part. The electrical engineering discipline is neglected. The Requirement Functional Logical Physical (RFLP) approach is a specific V-model-derived method particularly adapted to design of mechatronic systems. In this method, the descending branch of V-model is divided into four specific phases: Requirement engineering, Functional design, Logical design and Physical design—RFLP (Lefèvre et al. 2012). As for the four phases, different technical tools can help the designers to realise the micro-level process. For example, the method APTE[1] and the language SysML have been used for the Requirement engineering, the method IDEF[2] and the language SysML for Functional design, Matlab/Simulink,[3] Modelica,[4] Bond Graph[5] and VHDL-AMS[6] for Logical design, Flux 3D,[7]Abaqus/CATIA[8] for detailed Physical design (Penas et al. 2011). Nowadays, the RFLP approach has been implemented as a basis for Systems Engineering in the Product Lifecycle Management (PLM) environment of Dassault Systèmes and can therefore be considered as a commercial approach (Kleiner and Kramer 2013). The drawback of the RFLP approach, however, is that the exchange of information between software design and other disciplines remains a challenge.

In summary, the V-model and its variants bring forth great benefits for proposing an effective way of representing a macro-level process for the design of complex systems (Department of Transportation 2007), but not all of them cover all the disciplines for the design of mechatronic systems (Bathelt et al. 2005; Penas et al. 2011; Kleiner and Kramer 2013). Moreover, they seldom or never pay attention on the micro-level process (Gausemeier and Moehringer 2003; VDI 2206 2003; Vasić and Lazarević 2008; Hofmann et al. 2010; Gausemeier et al. 2011; Nattermann and Anderl 2013; Barbieri et al. 2014). Finally, to the authors' knowledge, none of current variants propose an effective method to ensure the consistency and traceability among different design phases.

Next subsection will present the state of the art of the hierarchical design model, which is used as the micro-level design process in the proposed design methodology.

### 2.3 Hierarchical design model

Hierarchical design model describes the structure in which a model is broken down into smaller parts. It helps the designers to describe product models from different viewpoints and guarantee the consistency of these models in the overall product development process (Hehenberger 2014).

According to VDI 2206 (2003), the development process of mechatronic systems can be carried out by using the V-model as macro-level process. After analysing the requirements for the whole system, the sub-functions are defined. The sub-systems (or components) are then to be developed simultaneously by the cooperating development teams. After verifying the sub-functions of the sub-systems (or components), they are integrated step by step. Then, the performance of the integrated system is checked.

The requirements of a new mechatronic system should be analysed, and a requirements specification (i.e. requirements list) should be determined once the design process carries out. However, identification of these requirements is often difficult. Seyff et al. (2009) propose a promising approach by applying use cases that incorporate possible scenarios to identify the product requirements. The initial requirements can be decomposed into further sub-requirements, thus creating a hierarchy of requirements. Therefore, a hierarchical structure is necessary to specify the requirements.

After the requirement definition, the mechatronic system's overall function, its most important sub-functions and their interactions should be determined, which leads to a functional structure. This functional structure can be organised hierarchically in order to describe the different levels of abstraction. Pahl and Beitz (1988) propose a method to show that a function structure can be developed by decomposing an overall function into sub-functions. Umeda et al. (1990) propose a way to construct the functional structure. They claim that function decomposition continues until the function matches a design in the catalogue. Stone and Wood (2000) formalise the modelling techniques of the functional basis for design based on previous studies. They propose that functional modelling of a device is an important step in the design process in which the focus is on the flows of material, energy and signals.

The system's architecture is formed by grouping the functions which are already collected before. The system's

[1] http://methode-apte.com/.
[2] http://www.idef.com/.
[3] http://www.mathworks.com.
[4] https://www.modelica.org/.
[5] http://www.bondgraph.org/.
[6] http://www.eda.org/.
[7] http://www.cedrat.com/fr.html.
[8] http://www.3ds.com/products-services/simulia/products/abaqus/.

structure can be decomposed hierarchically into sub-systems, components and the interfaces among them. van Beek et al. (2010) propose a method based on the FBS modelling. In this method, the three steps of decomposition into components, identification of the relations between the components and clustering of the components into modules are realised.

The literature review of the hierarchical structures for the requirements design, the functional design and the architectural design shows that the hierarchical structure can be used as an effective support for the micro-level process. The consistency among different design phases can be also ensured by such hierarchical structure. A multidisciplinary interface model is devoted to guarantee the consistency between the macro-level process and the micro-level process. Next subsection will review the studies on interface model developed in current product models.

### 2.4 Interface model

The topic of interface is at the heart of the multidisciplinary nature of Systems Engineering (Fosse et al. 2013). From the mid-1980s, the interface between systems and sub-systems has been widely used in software engineering (Dorfman 1990; Hoffman 1990). During the design process of software, separated module of a program executes one aspect of the desire functionality. Such modules interact with each other through interfaces. As system became increasingly complex, a complex system is always decomposed into sub-systems, components and interfaces. Interface management is considered as one of the most powerful tools of systems management because the interfaces are used to manage the interdependencies between different models (Blyler 2004). The paper will review the previous studies on interface model by focusing on the current product models.

STandard for the Exchange of Product model data (STEP) is actually a series of standards, known as ISO 10303 developed by experts' worldwide (ISO10303-1 1994; Pratt 2001). The parts known as APs (Application Protocol) of STEP are used to define the scope, context and information requirements of applications. STEP AP233 describes the key product data and information for systems engineering that must be exchanged between dissimilar applications for requirements engineering and for systems modelling and simulation. Industries that can benefit from using AP233 are automotive, aerospace, shipbuilding, consumer goods electronics and others with complex products and processes (ISO10303-233 2012). In AP233, an interface connector is defined as the term for the part of a system that interacts with other systems or the environment, and the interface connection as the link between connectors (Sellgren et al. 2009), but no more details of the interface connector and interface connection are provided by AP233. Pandikow et al. (2000) try to integrate the UML with AP233 in order to provide detailed interface and association descriptions to extend AP233, but this extended AP233 focuses on the software engineering, not on the mechatronics engineering. Core Product Model (CPM), an abstract model with generic semantics, initially developed at NIST (National Institute of Standards and Technology), can support a full range of PLM information (Fenves et al. 2006). Nevertheless, the interfaces model is not developed in CPM. Some extensions of CPM have been proposed to address the interface issue. For example, Zha et al. (2005) develop an extension of the CPM, called Embedded System Model (ESM). Like the term "interface connector" of STEP AP233, "Port" is defined to describe the connection point of interface in ESM. The extended model provides the interface features between hardware/software, hardware/hardware and software/software. To a certain extent, an embedded system is similar to a mechatronic system and ESM partially performs the collaboration between electronic and software disciplines, but the collaboration with mechanical discipline has not been deeply discussed.

Product-Process-Organisation (PPO) model describes information of product, process and organisation (Noël and Roucoules 2008). An interface class is described in the product model by the way a component (mechanical, electrical, etc.) may be linked to another (Nowak et al. 2004), but the interface class is simply divided into Common Interfaces (CI), Alternative Interfaces (AI) and View Interfaces (VI) and its details has not been given. As shown with recent PPO model developments, PPO is generally considered as an extensible model (Le Duigou et al. 2011). Therefore, the extension of PPO for detailing the interfaces in mechatronic systems should be developed.

A multidisciplinary interface model has been defined through the past continuous research efforts of the authors (Zheng et al. 2015a, b, 2016). The authors propose the multidisciplinary interface model to deal with the problem of multidisciplinary integration. Figure 2 shows the multidisciplinary interface model specified with a UML class diagram. This multidisciplinary interface model not only provides a structured representation to store the design data related to the interface in order to achieve the knowledge reuse, but offers a compatibility rules to the designers to guarantee the different components integrate correctly.

First, the proposed multidisciplinary interface model provides a structured representation to store the design data related to the interface in order to realise the knowledge reuse. It contains classes to define the attributes of one interface and its ports. The interface attributes are defined by taking into consideration of three different features: **type**, **configuration** and **desired/undesired**. **Type** attribute

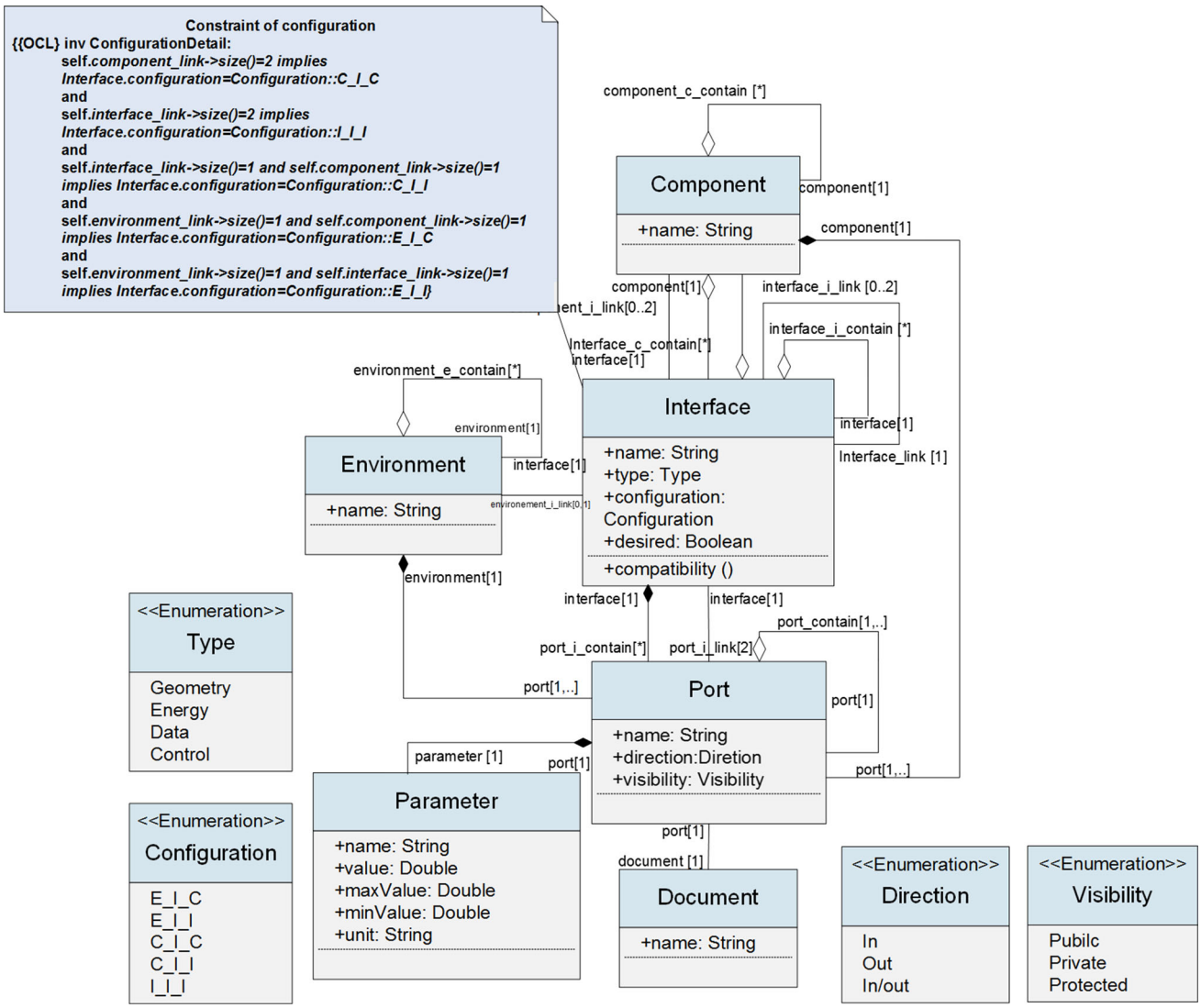



**Fig. 2** UML class diagram of multidisciplinary interface model

focuses on which types of transfer (geometric, energy, control or data) occur through one interface. **Configuration** attribute describes which elements are linked by the interface. **Desired/undesired** attribute expresses whether the interface creates positive effects (e.g. data or energy transmission) or unintended side effects (e.g. heat, magnetic fields, vibration and other side effects). In summary, the attributes contained in the multidisciplinary interface model provide a common representation for the interfaces defined by design teams of different disciplines. The term "port" is considered as the primary location through which one element of a system interacts with other elements. Figure 2 shows that there are two classes linking with class **Port** to describe the port's details. The class **Parameter** specifies the principal parameters related to the port which can be quantified, such as the wheel size, the input impedance or the image resolution, while the class **Document** is used to store the documents (such as standards, special constraints, etc.) related to the port which cannot be simply quantified.

Second, the multidisciplinary interface model offers the compatibility rules to the designers to guarantee the different components integrate correctly. The method **compatibility()** is contained by the class Interface to check the compatibility of the interface. One example is cited here to illustrate the compatibility test method. Two components (Component 1 and Component 2) are connected by an interface (Interface) through the ports (CP1 and CP2). Two foundational compatibility rules are presented as follows.

***Rule 1:***

***CP1. Parameters1. value = CP2. Parameters2. value***
***CP1. Parameters1. unit = CP2. Parameters2. unit***

***Rule 2:***

***CP1. Parameter1. value < CP2. Parameters2. maxValue***
***CP1. Parameter1. value > CP2. Parameters2. minValue***
***CP1. Parameter1. unit = CP2. Parameters2. unit***

***CP1. Parameter1*** represents the parameter stored in the class **Parameter** of the port **CP1**, and ***CP2. Parameter2*** is the parameter of port **CP2**. In the compatibility **Rule 1**, in order to ensure the two components integrate with each other correctly, both the value and the unit of the parameters of **CP1** and **CP2** should be equal. However, sometimes the design parameter of one port is not specified by an exact value accurately. The compatibility **Rule 2** is used to illustrate that case. If the port **CP2** specifies the parameter by using an interval (***minValue***, ***maxValue***), the parameter of ***CP1*** should satisfy that ***CP1. Parameter1.value*** $\in$ ***(CP2. Parameters2. minValue, CP2. Parameters2. maxValue)***.

The rules previously presented are considered as two foundational rules to deal with the two types of interface compatibility. The designers may encounter some complex compatibility problems. For example, the parameter of port CP1 can be equal to several values at the same time, or it can lie in several intervals which are discontinuous. Such complex compatibility rules can be considered as the combination of the two foundational compatibility rules, which are combined with the logical operation ("and" or "or").

Nowadays, the increasing integration of mechatronic systems shows the trend to transfer several different types of transfers through one interface. For instance, as for the interface between two electrical components, the electrical energy (voltage) can be transferred through this interface, and meanwhile, the geometrical connections (pin numbers) exist between the two components in order to have a better physical integration. The technology of power-line communication (PLC) is another example. The PLC is used to carry data among conductors that are also used simultaneously for AC electric power transmission or electric power distribution. In the example of the PLC, data and energy can be transferred simultaneously through one interface. Such interfaces should be further decomposed and refined into sub-interfaces according to the different transfers through them, and the compatibility of each interface should be tested once the interface models have been instantiated.

The multidisciplinary interface model is an effective support for the design of mechatronic systems, but one drawback of the multidisciplinary interface model is that it cannot be used directly to manage the design process. In the paper, the multidisciplinary interface model is integrated with the proposed methodology to support of the design process of mechatronic systems; therefore, the drawback of the multidisciplinary interface model can be eliminated. Next section presents the details of the proposed design methodology for the design of mechatronic systems.

## 3 Design methodology based on interface model

The design methodology presented in the paper adopts an extended V-model as the macro-level process and the hierarchical design model as the micro-level process. The multidisciplinary interface model helps the designers to ensure the consistency between the two levels. The details of the proposed design methodology will be presented hereafter.

### 3.1 Macro-level process

The macro-level design process adopted an extended V-model to present the general flow for the design process of mechatronic system. The left branch of the extended V-model represents the system design sub-process and is described with qualitative models. After analysing all requirements for the whole system, the sub-functions and sub-systems are defined. Three design phases are identified during the system design sub-process: Requirements specification phase, Functional model phase and Architectural model phase (see Fig. 3).

The discipline-specific design sub-process is presented at the bottom of the extended V-model. The objective of the discipline-specific sub-process is to obtain the physical elements of the system such as hardware components or software code. The sub-systems have a very discipline-specific character and are developed simultaneously by the different design teams. The models constructed by the teams from different disciplines are quantitative for the most part.

The right branch of the V-model represents the system integration sub-process. In the extended V-model, this sub-process can be divided into two phases: Compatibility test phase and Verification and validation phase. The objective of the Compatibility test phase is to guarantee the sub-systems integrate correctly and to ensure the multidisciplinary integration among different design teams. If the sub-systems prove to be incompatible with each other, an iterative process should be carried on. This compatibility test in the early phase of system integration will greatly reduce the iterations number in the later phase so that the overall development costs and the time-to-market can decrease accordingly. The Verification and Validation phase is used to test the performance of the integrated system and check whether the system realises the proposed function and satisfies all the requirements proposed before. If the system has to be improved, the previous design phases will be repeated.

In summary, the macro-level design process is developed based on V-model, which is considered as one of the

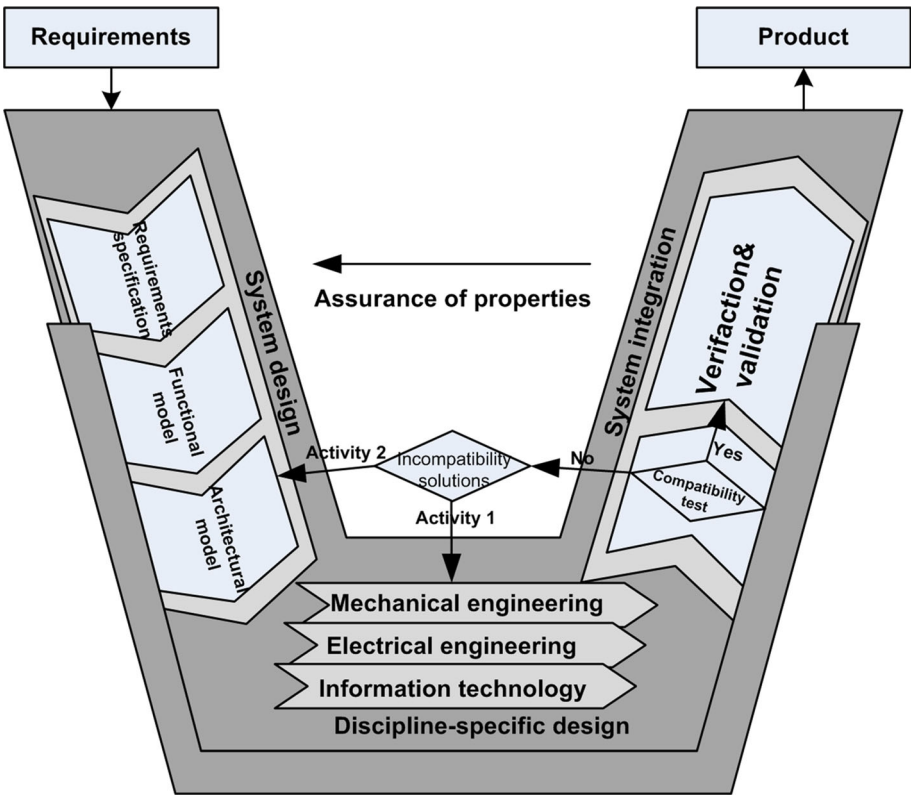



**Fig. 3** Macro-level process: an extended V-model based on VDI 2206

more typical systems engineering approaches. Therefore, the three main design sub-processes (system design sub-process, discipline-specific design sub-process and system integration sub-process) of macro-level design process are sequentially organised in line with the systems engineering practice. However, designers from various disciplines carry out concurrent design activities during each sub-process of the whole design process in order to achieve a more integrated design. Organisation of design activities during each sub-process is called micro-level design process. Next subsection presents the micro-level design process supported by the hierarchical design model in detail.

### 3.2 Micro-level process

In this section, the micro-level process will be discussed in more detail. The hierarchical structure will be applied in the three design phases of the system design sub-process. A hierarchy of the design parameters is also proposed to help the designers to define the key parameters in the discipline-specific design sub-process.

#### *3.2.1 Requirement specification*

The requirements specification is derived from all requirements on the mechatronic system, and these requirements can provide initial information about what is required by the customers. Requirements can be applied to the overall system, every single sub-system (or component) and the interconnection between two sub-systems. Therefore, requirements can be detailed by decomposing into further sub-requirements, thus creating a hierarchy of requirements.

The requirements can be classified into several groups: global requirements, cumulative requirements, specific requirements and interconnected requirements. Examples will be taken to clarify the four groups. The global requirement, for example, is that certain materials must not be used in any components of the mechatronic system. An example of a cumulative requirement is the size of the mechatronic system, to which every component contributes to a certain degree. A good example of specific requirement is electromagnetic pulse protection, which is only relevant to the electrical system, or the lubrication which applies only to moving components. The interconnected requirements are the requirements, which may be influenced by other requirements. An example of interconnected requirement is the performance of the system, such as the energy consumption (Requirement 1), which relates to the dynamical properties of the system and describes the performance of the system. This requirement is influenced by the maximum mass of the systems (Requirement 2) and the required force (Requirement 3). In other words, the Requirement 2 and 3 can be determined separately;

however, the energy consumption should be determined by considering the system's mass and requirement force according to the physical relationships. In this example, Requirement 1 is influenced by Requirements 2 and 3 and named "interconnected requirement" (Hehenberger 2014).

Above introduction reveals that two kinds of relationships between requirements should be manipulated in the requirement specification phase. On the one hand, the relationship between the requirement and sub-requirement can be described as "composition" when requirements are detailed by decomposing into sub-requirements; on the other hand, the relationship "interconnection" is proposed so that the designers can easily find the details which reflect additional implementation considerations or constrains between two interconnected requirements. A hierarchical structure is necessary in order to manipulate the proposed four groups of requirements as well as the two kinds of relationships between them. Many mainstream modelling languages (e.g. SysML) can be used to support such hierarchical structure and to implement the relationships for requirement specification.

Once the requirements are placed in groups, they can be considered accordingly in specific sub-functions. For example, the breakdown voltage, one sub-requirement of security requirements, should be considered to avoid the breakdown condition (sub-function). Therefore, the functional model can be constructed according to the hierarchical structure of requirements specification.

### 3.2.2 *Functional model*

A functional model refers to the phase of modelling and specification of the functional solutions. Functional model plays a significant role during the system design process because it is constructed as a bridge between the customers and the mechatronic system. On the one hand, the functions and sub-functions proposed in the functional models are used to satisfy the customer's requirements. On the other hand, the system architecture is decided based on this functional model.

In this design phase, the functional model of mechatronic systems should be created. It must fulfil the specified requirements and thus provide the basis for deriving the functional structure. Hierarchical structure can also be used for functional modelling. If it is assumed that a complex mechatronic system comprises a certain number of elementary functions, the functional structure as cooperation among these elementary functions should be taken into consideration. A single elementary function is characterised by using primarily one clearly defined effect (e.g. physical, chemical or biological), which may be regarded as indivisible within the set of functions. An architectural model is then formed by grouping sub-functions from the functional model in order to implement the proposed functions.

In the proposed design methodology, the functional model is used to model and specify the functional solutions in a hierarchical structure. Although a number of modelling techniques have been proposed to help the designers to define the functional structure, such as the method APTE and IDEF, the language SysML and the tool like 3DEXPERIENCE[9] of Dassault Systèmes, however, not all the modelling techniques above mentioned can represent such hierarchical structure. In the paper, the Functional **View** of the 3DEXPERIENCE platform is used to represent functions.

### 3.2.3 *Architectural model*

After the hierarchical decomposition process of function, the designers should find the sub-systems, which can embody the proposed sub-functions. In other words, these sub-systems should exhibit decomposed sub-functions so that there is the consistency between the functional model and the architectural model. A complete architecture will then be constructed by decomposing the sub-systems in architectural model phase.

The decomposition process should be applied recursively. However, there is always a research question, that is, what are appropriate hierarchies and granularities for architectural models of mechatronic systems? In order to answer this question, the authors two steps to help the designers decompose the systems. In the proposed architectural hierarchy, the system can be presented top–down as mechatronic system, mechatronic module and discipline-specific component. A mechatronic module is defined as a mechatronic sub-system at the lowest hierarchical level of mechatronic system and is indivisible within the set of mechatronic sub-systems. "Indivisible" means that a mechatronic module can be decomposed only into discipline-specific (non-mechatronic) components, but not into other mechatronic modules or mechatronic system components. Discipline-specific components are considered as the lowest level of the system's architecture. "Lowest level" refers to the components that can be obtained with standard components, past designs or within the discipline-specific teams.

In the proposed two-step decomposition process, the mechatronic systems are decomposed into mechatronic modules in the first step. In this step, designers should primarily consider their design experience. For example, can the mechatronic module be implemented based on past design or standard component in handbooks? If certain mechatronic modules can be perfectly realised thanks to

[9] http://www.3ds.com/about-3ds/3dexperience-platform/.

earlier designs or with standard components, further decomposition is not necessary. This situation implies that it is not necessary to decompose a system down to its discipline-specific components. However, such past designs or standard components are not always available. For those cases, the second step of the decomposition process is proposed. In this step, the mechatronic module is further decomposed down to discipline-specific components that can be obtained with standard components, earlier designs or via discipline-specific teams (Fig. 4). Each model pillar in Fig. 4 characterises the group of discipline-specific components, which is structured into several hierarchical levels corresponding to the proceeding degree of detailing. During the decomposition process, the interfaces among the discipline-specific sub-systems should be also clarified and decomposed in order to verify whether the sub-systems can be correctly integrated with each other.

As discussed before, pillar design model requires the designers to decompose the mechatronic system into mechatronic modules. If past designs or standard components are not available for certain mechatronic modules, such modules should be further decomposed into discipline-specific sub-systems. During this decomposition process, the interfaces linked to the mechatronic modules or even discipline-specific sub-systems should be simultaneously decomposed (see Fig. 4). Attention should be paid to the traceability of the interfaces during the decomposition process. In other words, the information stored in the product models needs to be unambiguously understood by the designers not only in different disciplines, but the different decomposition levels as well. The multidisciplinary interface model proposed by (Zheng et al. 2016) provides a common representation of interfaces by using three attributes to describe an interface of mechatronic systems and helps the designers to overcome the lack of commonality in interface definitions from different decomposition levels or different disciplines. Before entering the discipline-specific design process, the consistency between each two design phases should be checked because functional models and architectural models are always created and maintained by the designers from different disciplines. As presented before, during the system design process, the functions and sub-functions proposed in the functional models are used to satisfy the costumer's requirements, while the architectural model for the sub-systems is constructed to embody the proposed sub-functions (Fig. 5). Therefore, it is necessary to ensure correctness of such models. Every requirement should be met by one or more functions (or sub-functions), and every sub-function must be realised by one or more sub-systems (in certain case several sub-functions can be realised by one sub-system).

After the mechatronic modules are further decomposed into discipline-specific sub-systems and the interfaces among them, these discipline-specific sub-systems (with their interface) will be developed by the engineers in different disciplines during the discipline-specific design phase.

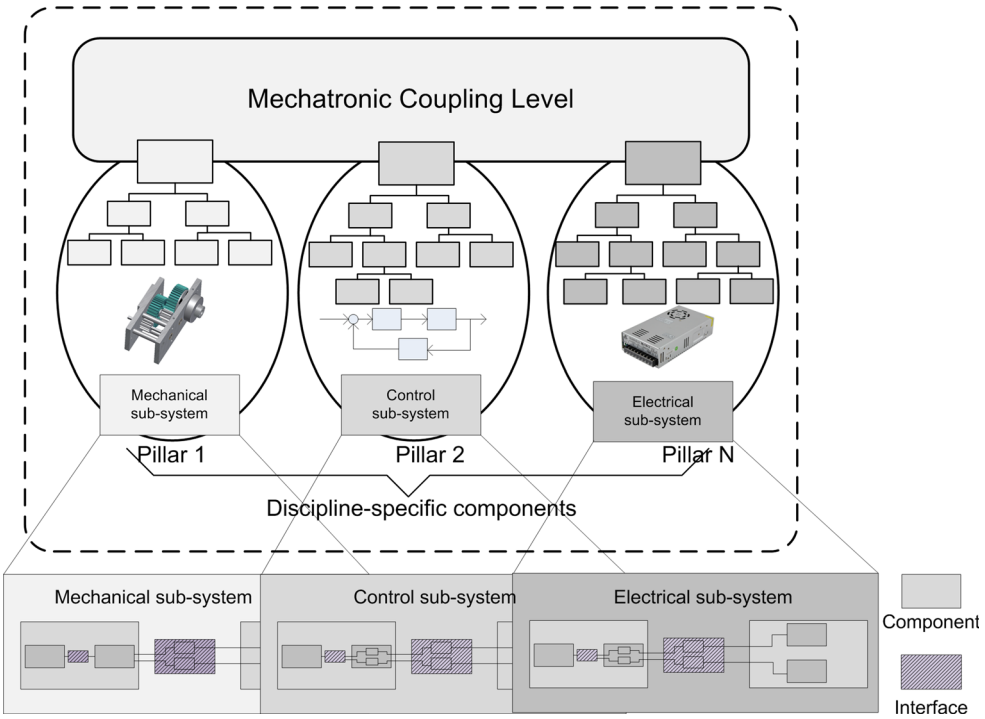



**Fig. 4** Mechatronic module

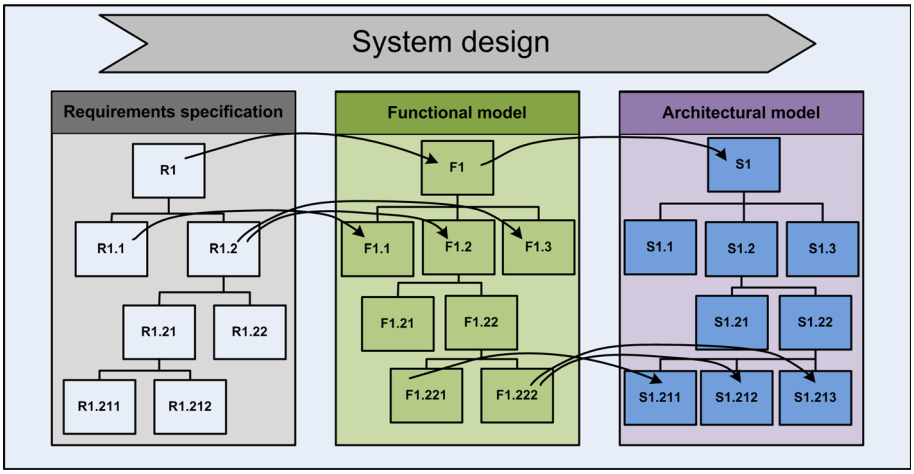



**Fig. 5** Consistency between different design phases during system design process

### 3.2.4 *Discipline-specific design*

During the discipline-specific design phase, the designers of each design discipline should develop sub-systems, which have been decided in the architectural model phase. Certain components of the discipline-specific sub-systems can be perfectly realised by the past designs or the standard components. However, if they cannot be realised by any past designs or standard components, a new design embodying the function of this component should be carried on.

As for the components which need to be designed as new embodiments to realise the function proposed before, the authors propose a hierarchy of the design parameters to help the designers to define the key parameters at an early stage of the discipline-specific design sub-process.

Figure 6 shows the hierarchy of the design parameters. The steps to establish such hierarchy of the design parameters will be described as follows:

- Step 1: Define the functional requirement (FR) of each model pillar by several design parameters (DP).
- Step 2: Define the FRs at level i + 1 by considering the FR at level and its DPs.
- Step 3: Classify the design parameters at one level into two categories, the internal design parameters and the external design parameters. The external parameters affect other parameters of the next level, while the internal parameters are exclusively local at the active level for dimensioning the component at this level, and they have nothing to do with other parameters.
- Step 4: The process of defining hierarchical levels must be repeated until well-known DPs (e.g. proven solutions and standard components) are achieved.
- Step 5: Iteration and feedback across hierarchical level should be carried on to verify whether the design parameters meet the functional requirement in each level.

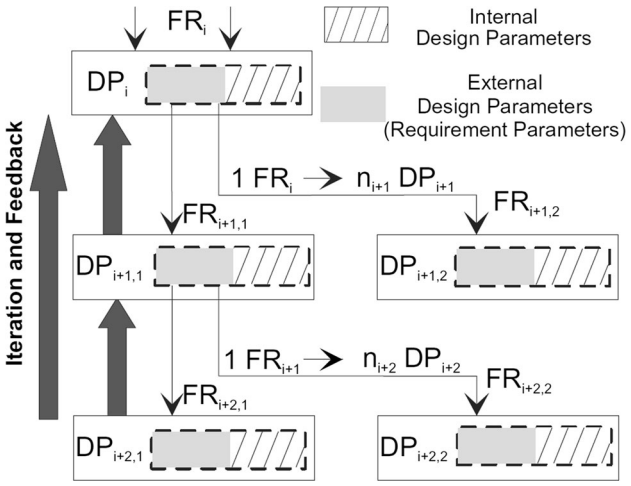



**Fig. 6** Hierarchy of design parameters (Hehenberger et al. 2010)

More information about the hierarchy of the design parameters can be found in (Hehenberger et al. 2010).

### 3.2.5 *Compatibility test and Verification and Validation*

The design of mechatronic systems requires a multidisciplinary collaboration, which often leads to the iteration during the concurrent design process, because the designers often have to jump back one or more steps to redesign or to tune what they created before to satisfy the parts designed by other designers. Therefore, the interface compatibility is very important for the mechatronic system, as it can detect the design errors before the design is completely finished, which can greatly reduce the unnecessary iterations. As a result, the overall development costs and the time-to-market can be decreased accordingly.

The current product models discussed in Sect. 2.4 show that attention has been paid to the port, which is considered as the primary location through which one component of a system interacts with other components. The main function of one interface is to connect two separated components and to exchange the information between them, so one interface can link two ports. As discussed in Sect. 3.2.4, the external parameter affects other parameters; therefore, it is stored in the port's class of the multidisciplinary interface model, which has been introduced in Sect. 2.4. The method **compatibility()** is contained by the interface class to check the compatibility of the interface, which helps the designers to guarantee the different components integrate correctly and eventually ensure the multidisciplinary integration among design teams. Once the model of an interface has been instantiated, the designers should check the compatibility of the interface by making use of the method **compatibility()**.

Two solutions are proposed to deal with the incompatible interface. These solutions will be presented as follows:

- Solution 1: Change one of the two component linked by the interface; the compatibility should therefore be checked again. This solution is called "component change".
- Solution 2: Decompose the interface into an interface–component–interface structure; the compatibility of the two newly created interfaces should be checked. This solution is called "interface decomposition".

Figure 7 illustrates these two kinds of solutions. In this example, a simple mechatronic system (System) is decomposed into two components (Component 1 and Component 2) and an interface (Interface 1). However, when the designs of the two components have been finished by the designers, the compatibility test result of the interface (Interface1) indicates that both components are incompatible with each other (Fig. 7a). Figure 7b shows the Component change solution. The Component 2 can be redesigned and replaced by the Component 3. Because one component should be redesigned, the design process will go back to the discipline-specific design sub-process. The compatibility of the interface (Interface 1) should be then checked again. Figure 7c shows the second solution–interface decomposition solution. A new component (Component 3) can be added between the two components, and two new interfaces (Interface 1.1 and Interface 1.2) will be created accordingly. Due to the decomposition of the incompatible interface, the design process will return to the architectural model phase. The compatibilities of the two new interfaces should be checked.

Compared with the interface decomposition solution, the component change solution is much simpler to operate by the designers, because the interface decomposition solution demands the designers to further decomposed the incompatible interface into an interface–component–interface structure and the architectural model of the entire mechatronic system should be modified (a new component is added into the mechatronic system). However, the interface decomposition solution is more often adopted during the design process. The first reason is that it can help the designers to refine the architectural model of the mechatronic system. In most cases, the architectural model of a mechatronic system cannot be completely decided by the designers during the system design sub-process due to some reasons, such as the lack of design experiences and the unforeseen incompatibility between two sub-systems. When the interface decomposition solution is adopted, the design process should go back to the architectural model phase. Therefore, the interface decomposition solution can be used as an effective support to help the designers to refine the architecture of the mechatronic system. The second reason is that the interface decomposition solution can help the designers to avoid the design conflict. In a complex mechatronic system, one component may link to others through several interfaces. If the component change solution is adopted by the designers to solve the incompatibility problem, other interfaces linked to this component which proves to be compatible before may become incompatible after the change of such component. Such conflict may always exist during the design process. The interface decomposition solution does not create design conflicts because the new component does not affect other components of the system.

If all the interfaces in the mechatronic system have proved to be compatible, which means that all the components has been correctly integrated with each other, the design process should move on to the verification and validation phase. The verification and validation phase is used to test the performance of the integrated system and check whether the system realises the proposed function and satisfies all the requirements proposed before. If the system has to be improved, the initial operation phase will be repeated. The above subsections, respectively, introduced the macro-level and micro-level process of the proposed design methodology. Next subsection will present the realisation of consistency between the two levels.

### 3.3 Realisation of consistency between macro- and micro-level design processes

In previous subsections, the authors introduce the extended V-model and the hierarchical design model used as the macro- and micro-level design processes, respectively. How to use the multidisciplinary interface model to support the refinement of system's architecture during the architectural model phase and to test the compatibility between

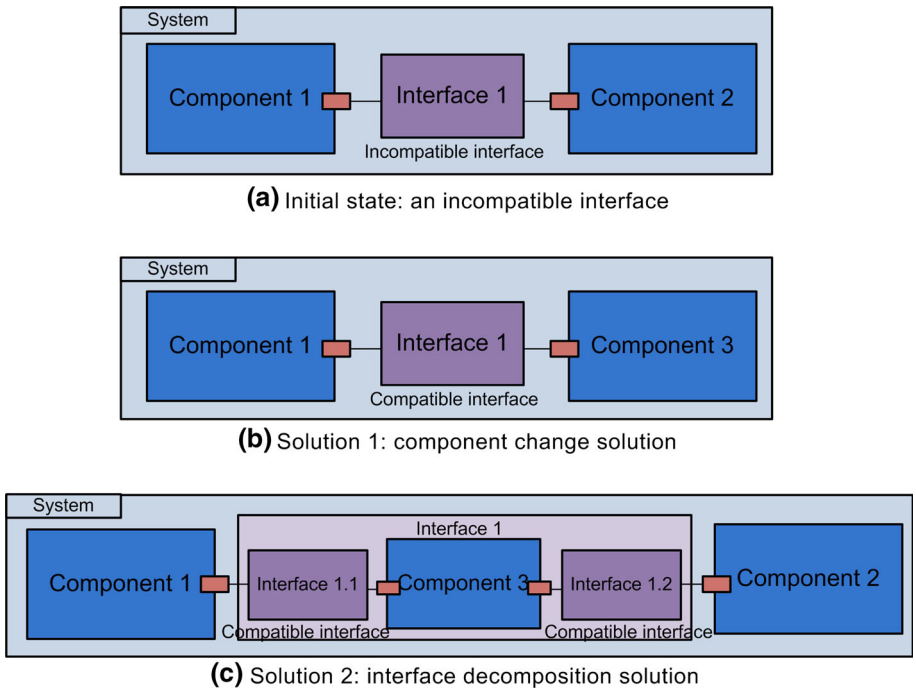



**Fig. 7** Example of solutions to incompatible interfaces

sub-systems in the compatibility test phase is also presented. This subsection will present the realisation of consistency between the two process models.

### 3.3.1 *From macro-level process to micro-level process: hierarchical structure*

The proposed extended V-model presents a general flow for the design process of mechatronic systems. It can be generally divided into three sub-processes: system design sub-process, discipline-specific design sub-process and system integration sub-process. During the two design sub-processes (the system design sub-process and discipline-specific sub-process), every design phase can be detailed by a hierarchical structure as the micro-level process.

In the system design sub-process, hierarchical structure can be used to establish the requirement specification, functional model and architectural model. Moreover, a hierarchy of the design parameters has been proposed to help the designers to define the key parameters in the discipline-specific design sub-process. Generally speaking, the consistency between the macro-level process and micro-level process is ensured by the hierarchical structure during the two design sub-processes.

### 3.3.2 *From micro-level process to macro-level process: multidisciplinary interface model*

The multidisciplinary interface model provides a common representation for the interfaces, which can be used during both the architectural model phase of the system design sub-process and the compatibility test phase of the system integration sub-process.

During the architectural model phase, the three attributes of the multidisciplinary interface model can define an interface from different decomposition levels or different disciplines with a same terminology. The consistency between the architectural model of system design sub-process and the discipline-specific design sub-process can be therefore ensured by this common representation of interfaces.

During the compatibility test phase, the method **compatibility**() proposed based on the multidisciplinary interface model will test the compatibility among the sub-systems (or components) and help the designers to decide whether the design process can enter the next phase (the verification and validation phase) or it should go back to the previous design phase (the discipline-specific sub-process or the architectural model phase of the system design sub-process). In summary, with the support of the multidisciplinary interface model, the designers can go back to the general development flow of the macro-level process from the specific design activities of the micro-process. The consistency between the micro-level process and the macro-level process is ensured by the multidisciplinary interface model.

Figure 8 shows the consistency between the two level processes. As discussed previously, the consistency is ensured by the hierarchical structure during the system design sub-process and discipline-specific design sub-

process, while by the interface model during the system integration sub-process. Next section will present the design process of a quadrotor, one type of Unmanned Aerial Vehicles (UAVs), as a case study to demonstrate the design methodology.

## 4 Case study

The aim of this section is to demonstrate the use of the proposed design methodology in a mechatronic design process. The case study chosen to demonstrate the design methodology is a quadrotor. Quadrotor is a one type of the UAVs with four propellers and a fixed cross structure. Because the quadrotor is considered as a complex mechatronic system integrating synergistically the electrical/electronic system, mechanical parts, information processing and aerodynamic technology, the multidisciplinary integration is required during the design process. Next subsections will present the principles of the quadrotor.

### 4.1 Presentation of quadrotor

The quadrotor is one typical UAV, which can be simply represented with four propellers in a cross configuration (Fig. 9). The propellers usually have identical pitch blades and are symmetric about the centre of the cross. Each propeller is connected to the motor through the reduction gears, and the movement of the quadrotor can be controlled by changing the rotational speed of each motor through the control sub-system.

The following sections will present the design process of the quadrotor based on the proposed multidisciplinary methodology by an implementation using 3DEXPERIENCE platform of Dassault systems.

### 4.2 System design sub-process

The first phase of the system design sub-process is to create the requirements specification. The general functional requirement proposed by the customers is that "this quadrotor should move from one location to another". In order to satisfy this functional requirement, hierarchical structure of the requirement should be constructed to detail this general requirement in the corresponding phase of the micro-level process.

When the requirements specification has been finished, the macro-level process enters into the functional model phase. In the corresponding phase of the micro-level process, a hierarchical functional model should be proposed according to the requirements specification. Attention should be paid to the consistency between the requirements specification and the functional model, which means that every requirement should be realised by one or more sub-functions. For example, the requirement "Power should be

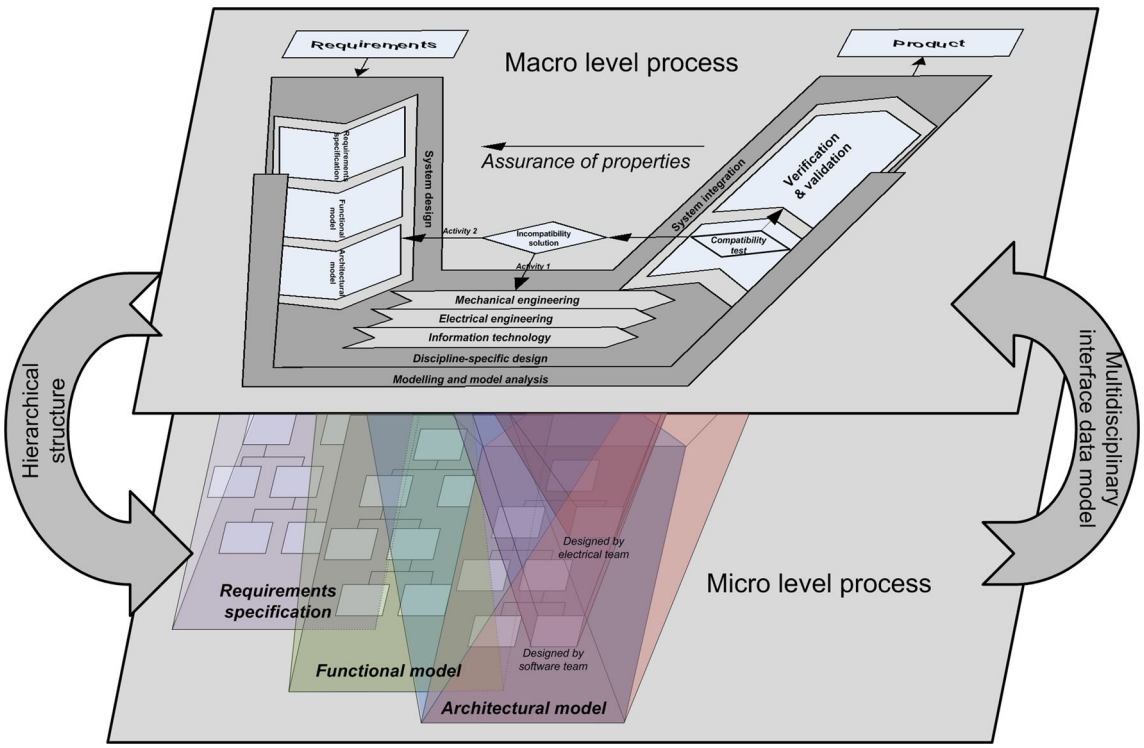



**Fig. 8** Consistency between two level processes

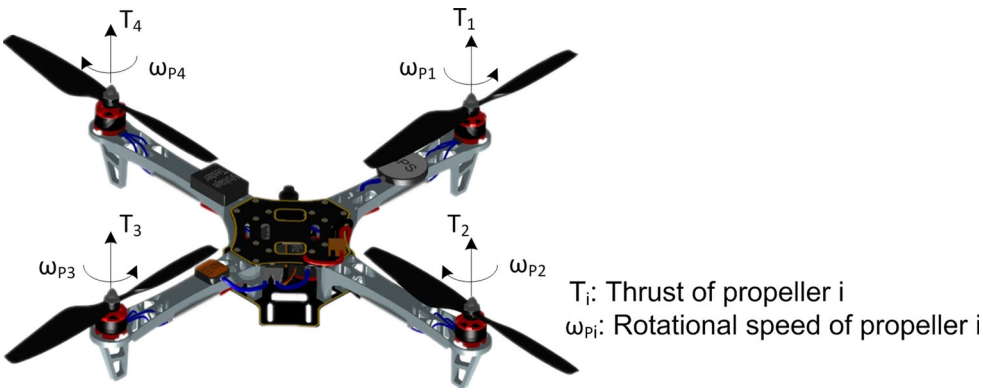



**Fig. 9** Simplified quadrotor representation

supplied to the quadrotor" can be realised by the sub-function "Supply the power".

The last phase of the system design sub-process is the architectural model phase. In the micro-level process, the system's general architecture formed by sub-systems or components can be obtained. The designers should firstly decompose the quadrotor into mechatronic modules and try to use the past designs or standard components to realise the functions of these mechatronic modules. For example, a DC motor can realise the function of providing the mechanical energy. If such standard components or the past designs do not exist, the designers can use the proposed pillar design model for the further decomposition. The interfaces among the sub-systems should be also clarified in this phase.

Figure 10 shows the hierarchical models in different design phases during the system design sub-process. In the paper, the authors use the 3DEXPERIENCE platform to support the management of the three hierarchical models and their consistency.

In order to clarify the interfaces among sub-systems of the architectural model, the Logical Editor workbench of 3DEXPERIENCE platform is used to describe the hierarchical structure. In Fig. 11, the sub-systems (or the components) and interfaces are represented, respectively, by grey and purple boxes.

### 4.3 Discipline-specific design

In the discipline-specific design sub-process, the hierarchy of design parameters is proposed to help the designers define the key parameters related to the quadrotor at an early stage. By analysing the architecture of the quadrotor decided previously, the system engineers find that the four main sub-systems of the quadrotor (i.e. the propellers driving sub-system, the control sub-system, the power supply sub-system and the propellers sub-system) should be developed by mechanical engineers, control engineers, electrical engineers and aerodynamics engineers separately. The propeller's rotational speed is considered as one of the most important design parameters because it plays an important role in all the involved disciplines. For example, the electrical engineers estimate the battery consumption by considering the propeller's rotational speed; one of the objectives of the control discipline is to control the propeller's rotational speed; the choice of the motor and the gearbox should take into account the propeller's rotational speed by the mechanical engineers; finally, the total aerodynamic forces are determined by the propeller's rotational speed. In order to demonstrate how the hierarchy of the design parameters helps the designer to define the key design parameter (i.e. propeller's rotational speed), the propellers driving sub-system (mechanical discipline), the propellers sub-system (aerodynamics discipline) and the interfaces between them are chosen as an example.

The propellers driving sub-system which is composed of the motor and the gear box is designed by the mechanical engineers. The design parameters related to the mechanical discipline should be taken into consideration. Hierarchy of the design parameters can be constructed according to the steps depicted in Sect. 3.2.4. Figure 12 presents the hierarchy of the design parameters for the quadrotor during the discipline-specific design process. Equation (1) at the first level of the hierarchy related to the mechanical discipline describes the general dynamics between the motor and the load (i.e. the gear box and the propeller). The design parameters of this level can be divided into internal design parameters and external design parameters. The motor's and the propeller's moments of inertia, the electric and mechanic motor constants, the motor resistance, the gear box efficiency and the gear box reduction ratio are classified as the internal design parameters, because they are exclusively local at this level and have nothing to do with the parameters related to other disciplines, while the motor's and propeller's rotational speeds, the load torque and the input motor voltage are considered as the external design parameters. Therefore, the process of defining hierarchical levels should be repeated until the internal design parameters are achieved. The load torque is chosen as an example to show the process of defining hierarchical

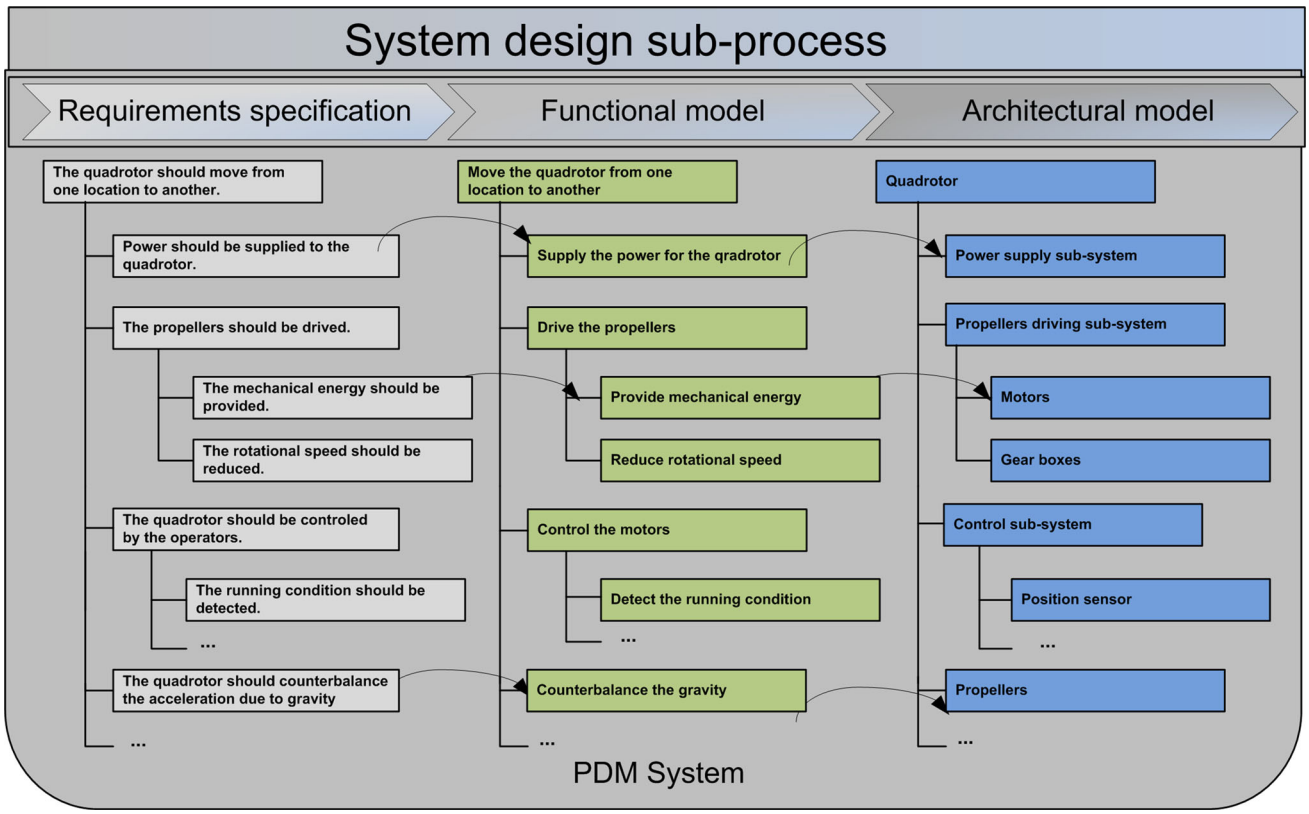



**Fig. 10** Hierarchical models and their consistency in the three design phases

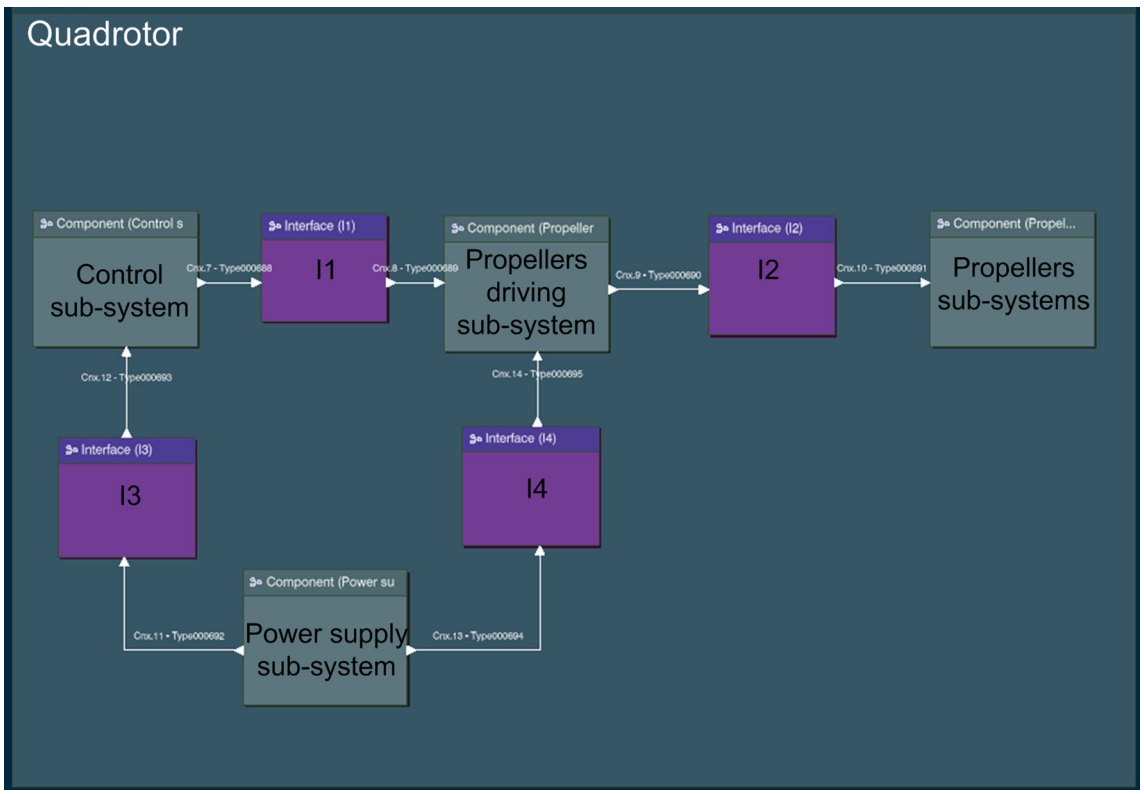
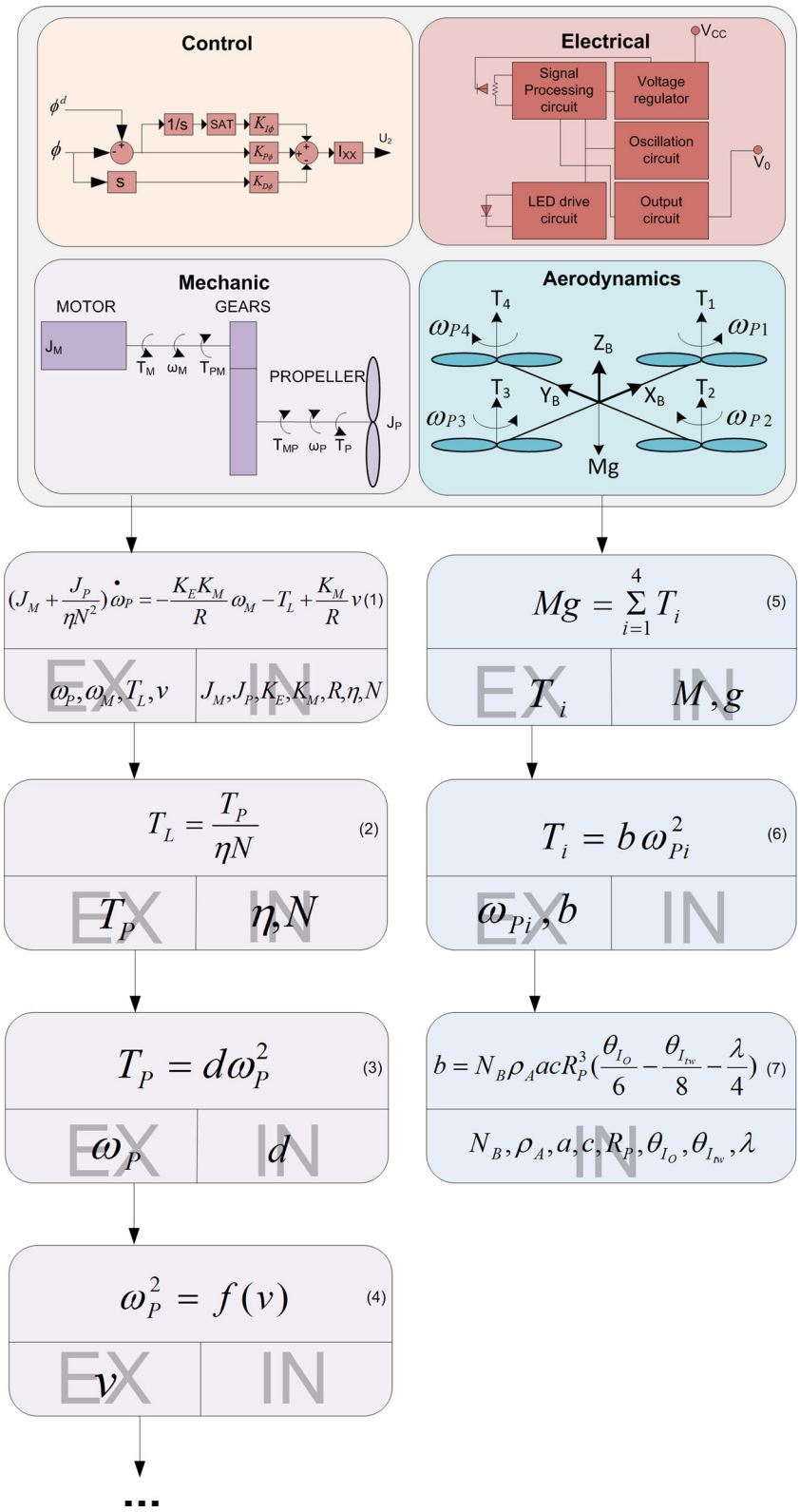



**Fig. 11** Representation of the quadrotor based on 3DEXPERIENCE platform




**$J_M$**: motor moment of inertia
**$J_P$**: propeller axis moment of inertia
**$T_M$**: motor torque
**$T_{PM}$**: propeller torque in the motor axis
**$T_{MP}$**: motor torque in the propeller axis
**$T_L$**:load torque
**$T_P$**:propeller torque
**$\omega_M$**: motor rotational speed
**$\omega_P$**: propeller rotational speed
**M**: quadrotor mass
**g**: acceleration due to gravity
**T**:vertical thrust of each propeller
**$K_E$**: electric motor constant
**$K_M$**: mechanic motor constant
**R**:motor resistance
**v**:input motor voltage
**η**:gear box efficiency
**N**:gear box reduction ratio
**b**: thrust factor
**Nb**: number of blades per propeller
**$\rho_A$**: air density
**a**: lift slope
**c**: average chord of the blade
**$R_P$**: propeller radius
**$\theta_{I0}$**: zero angle of incidence
**$\theta_{Itw}$**: twist angle of incidence
**λ**: inflow ratio

**Fig. 12** Hierarchy of design parameters related to the mechanical and aerodynamics disciplines

levels for the external design parameters by Eqs. (2), (3) and (4). Equation (4) specifies that the propeller's rotational speed to the input motor voltage is not linear, so it is much more effective and efficient to estimate the propeller's rotational speed from the measurements results of experiments than to obtain the analytic solution by solving

the differential equation. By referring to the data sheet of the DC motor, the mechanical engineers find that the motor's rated voltage is 9.6 V. The propeller's rotational speed at the rated voltage is 221 rad/s, which can be obtained from the experiment.

Another hierarchy of the design parameters related to the aerodynamics discipline is constructed by the aerodynamics engineers. Equation (5) at the first level shows that the thrusts created by the propellers should counterbalance the force due to gravity. Considering the external design parameters and internal design parameters, the aerodynamics engineers can construct their own hierarchy of design parameters. In this case study, the total mass of the quadrotor ($M$) is 1.0 kg, and the thrust factor ($b$) is 53.8 N $s^2$, which can be obtained from Eq. (7). Taking into account Eqs. (5), (6) and (7), the engineers can obtain the minimum rotational speed of the propeller ($\omega_{\text{pmin}}$) which can maintain the quadrotor on the flight state is 213 rad/s.

### 4.4 System integration

In the system integration sub-process, the designers should check the compatibility of the interface once the model of an interface has been instantiated by making use of the compatibility rules in order to achieve the multidisciplinary integration.

Section 4.2 introduces the discipline-specific design phase for the propellers driving sub-system and the propellers sub-system. The hierarchy of the design parameters shows that some design parameters are exclusively local at this level and have nothing to do with the parameters related to other disciplines (i.e. internal design parameters), while others affect the design parameters related to other disciplines (i.e. external design parameters). The propeller's rotational speed is considered as the external design parameter, and both the mechanical engineers and the aerodynamics engineers propose their own propeller's rotational speed. Therefore, the compatibility of the interface between the propellers driving sub-system and the propellers sub-system should be checked by using the compatibility rules of the interface model.

Previous subsection shows that the minimum rotational speed of the propeller to maintain the quadrotor on the flight state has been calculated by the aerodynamics engineers. However, the propellers driving sub-system, including the motor and the gear box, is designed by the mechanical engineers. The compatibility of the interface between the propellers sub-system and the propellers driving sub-system should be checked in order to ensure that the propeller's rotational speed provided by the motor at the rated voltage is greater than the minimum rotational speed. The interface compatibility rule is realised thanks to the Knowledgeware workbenches of 3DEXPERIENCE platform (Fig. 13). Figure 13a shows the instance of the multidisciplinary interface model for the interface (I2) between the propellers driving sub-system and the propellers sub-system. Figure 13b shows the compatibility rules and the test result. Because the propeller's rotational speed provided by the motor at the rated voltage is greater than the minimum rotational speed which can maintain the quadrotor on the flight state, the interface proves to be compatible. It should be emphasised that the energy interface related to the rotational speed of propellers is not the only interface between the propellers driving sub-system and the propellers sub-system, and other types of interfaces (such as the geometric interface) may also exist between the two sub-systems. In that case the energy interface is just one sub-interface of the interface I2. However, for the sake of clarity, the authors do not further decompose the interface I2 into sub-interfaces, but use the I2 to represent the energy interface.

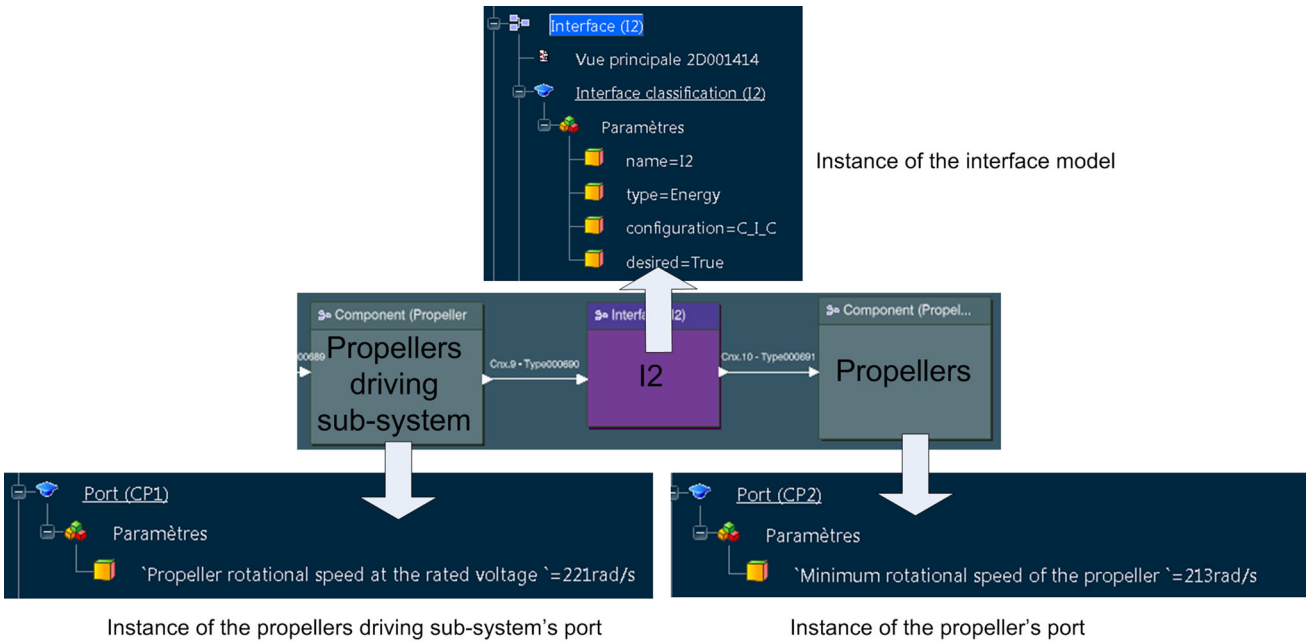
### 4.5 Synthesis on results of the case study

The quadrotor is used as a case study to demonstrate the proposed design methodology. The design process guided by this design methodology and its effectiveness will be summarised as follows:

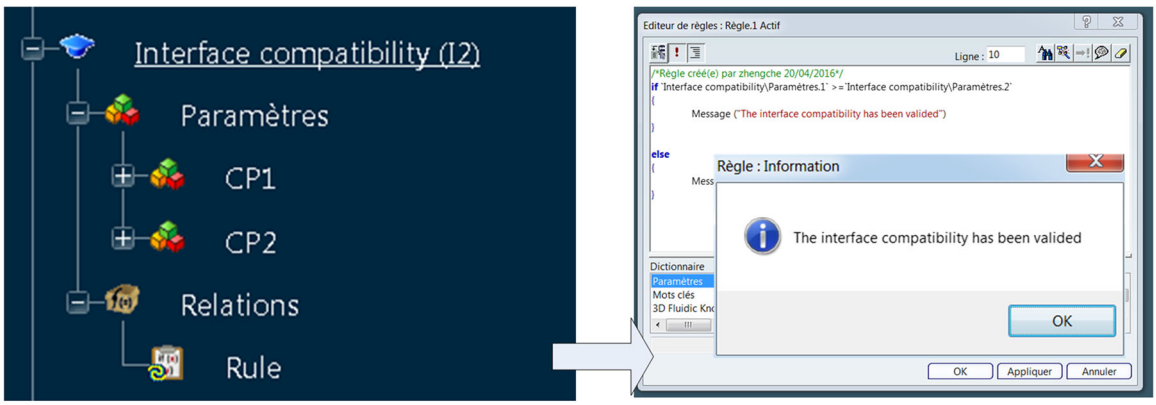
As for the system design sub-process, the main tasks of this design sub-process are to identify the customers' requirements, analyse system's functions and finally define the system's architecture. Therefore, three phases of the system design sub-process have been proposed in the macro-level process, including the requirements specification phase, the functional model phase and the architectural model phase. The system design sub-process of the quadrotor guided by the design methodology demonstrates that the proposed hierarchical structures can effectively guarantee the design consistency among the three phases.

In the discipline-specific design sub-process, the hierarchy of design parameters is proposed to help the designers define the key parameters related to the quadrotor at an early stage. In this case study, the propellers driving sub-system (mechanical discipline), the propellers sub-system (aerodynamics discipline) and the interfaces between them are chosen, and the hierarchies of the design parameters which help the designers to fix the propeller's rotational speed are demonstrated by the two involved disciplines. The demonstration indicates that the hierarchy of the design parameters is an effective solution to handle the multidisciplinary design. It is able to shift the design of mechatronic systems from the multidisciplinary level down to the discipline-specific level.

In the system integration sub-process, the compatibility of the interface between the propellers driving sub-system and the propellers sub-system is checked by the




**(a) Instance of the interface between the propellers driving sub-system and the propeller**




**(b) Interface compatibility rule and test result**

**Fig. 13** Interface model instance and compatibility test

compatibility rules of the multidisciplinary interface model. The multidisciplinary interface model provides a common representation for the interfaces. It can detect the design errors before the design is completely finished, which can greatly reduce the unnecessary iterations. As a result, the overall development costs and the time-to-market can be decreased accordingly.

## 5 Discussion

In the paper, the authors propose a design methodology adopting an extended V-model as the macro-level design process and the hierarchical design model as the micro-level design process. The multidisciplinary interface model is used to ensure the consistency between the two levels. Therefore, the three aspects, macro-level design process, micro-level design process and multidisciplinary interface model, are considered as the cores of the paper. The three aspects are coherent with each other and their combination help designers to achieve integrated design:

- **Macro-level design process**: An extended V-model is defined as the macro-level design process. In this design process, sub-processes and their design phases are presented. An iterative process is also described to guarantee the components integrate correctly and eventually to ensure the multidisciplinary integration among different design teams.

- **Micro-level design process**: The hierarchical design model is used to support the micro-level design process, which describes the details of design activities in every design phase.
- **Multidisciplinary interface model** The multidisciplinary interface model is proposed to ensure the consistency between the two levels of design process. With the support of the interface, the designers can go back to the general development flow of the macro-level process from the specific design activities, and therefore, the consistency between the two levels of design process is ensured.

The paper provides the design process of a quadrotor as a case study to demonstrate the design methodology for mechatronic systems, but this design methodology still has some limitations and needs to be further expanded in several directions:

- During the discipline-specific design sub-process, the authors propose a hierarchy of the design parameters to help the designers to define the key parameters of the components at an early stage. The authors provide an optimal model to describe this hierarchy of the design parameters (see Fig. 6). In this optimal model, every Functional Requirement can be satisfied by one component possessing several Design Parameters. However, due to the increasing integration level of systems or the designers' experiences, various hierarchies may be constructed by different designers when facing the same system. And even sometimes certain Functional Requirement should be satisfied by a systematic integration of numbers of components. Although a method called Degree of Mechatronic Coupling (DoMC) has been proposed by (Hehenberger et al. 2010) to help the designers to analyse and evaluate different hierarchies of the design parameters, more attention should be still paid to the issue of hierarchy in the future research.
- 3DEXPERIENCE platform allows the designers to construct the hierarchical structures for the requirement specification, functional model and the architectural model, and a demonstrator of the multidisciplinary interface model based on the Knowledgeware workbenches of 3DEXPERIENCE platform has been developed by the authors. However, an automatic IT platform means more than that. It should not only be able to facilitate the collaboration and communication among the design teams of different disciplines, which have been partially realised by the demonstrator presented in the paper, but also provide an automation of these design activities or tasks according to the decisions made by the designers. Therefore, such automatic IT platform which can fully support the entire design methodology should be further developed in future.
- Identification of all requirements on the total system, as the first design phase of the proposed design methodology, depends largely on the experience of the designers. Therefore, different designers may propose diverse architectural models which are derived from the requirements specifications. As for a designer who does not have enough design experiences, a complete architectural model cannot be obtained without iterations, and it needs to be improved along with the design process. As presented in Sect. 3.2.5, the second solution to the incompatibility problem–interface decomposition solution can help the designers to improve the architectural model by decomposing the incompatible interface. However, as for an experienced mechatronic system designer, a more complete architectural model can be fixed at the early stage of the system design sub-process and the iterative loop to solve the incompatibility problem can be reduced accordingly. Therefore, the designers' experiences play significant roles for the design of mechatronic system. Much more attention should be paid to the designers' experiences because the iterative loops can be greatly avoided so that the development costs and the time-to-market can be decreased.
- The V-model is considered as an effective way of representing the systems engineering and development process. The principle of V-model for the design of complex systems can be simply concluded as "Decomposition and definition (upper-left)" and "Integration and recomposition" (Department of Transportation 2007). Such "Decomposition-Recomposition" principle proves to be quite useful to support the design of complex systems. However, the real projects may deviate from the V-model. One challenge of V-model is that it is not flexible enough. The requirement documents may change due to the rapid changing market during the design process; therefore, the information, data or documents used for the design should be updated. Additionally, the V-model is proposed to solve the design problems of complex systems. Nevertheless, as for a simple system, or even a complex system but the design process of its similar system (i.e. product family) has been well established and management, it is not necessary to carry on the design process according to the V-model (interested reader can refer to (Kolberg et al. 2014); in this reference, a comprehensive robot development process has been presented, and it proves to be successful, but the development process does not follow the V-model and even the concurrent design approach has not been adopted by the process). Although sometimes the designers may follow the

design process which deviate significantly from the V-model as the macro-level process, however, the approaches in the micro-level process proposed in the paper, such as the hierarchical structure to specify the requirements, analyse the functions and define the architecture, the hierarchy of the design parameters and the multidisciplinary interface model can be still used to support the design process. The hierarchical structure can guarantee the design consistency among the different design phases, the hierarchy of the design parameters can help the designers to define the key parameters in an early stage, while the multidisciplinary interface model can greatly help the designers to avoid the design errors and reduce the iterative loops during the design process.

## 6 Conclusions

The paper has presented a design methodology which can aid the designers to achieve the multidisciplinary integration for the design of mechatronic systems. A case study—a quadrotor—is used to demonstrate the proposed design methodology by using 3DEXPERIENCE platform.

In order to achieve the multidisciplinary integration, the designers should pay attention to both the macro-level process and the micro-level process during the collaborative design process of mechatronic systems. The macro-level process describes the generic procedure for the design of mechatronic systems, while the micro-level process supports every specific design phase where individual designers can structure design sub-tasks and proceed and react in unforeseen situations. This proposed design methodology uses an extended V-model as the macro-level process. It describes the generic procedure for the design of mechatronic systems from the identification of all requirements on the total system to a user-validated system. The micro-level process is realised by the hierarchical design model. Such hierarchical design model supports each design phase where individual designers can proceed and react in unforeseen situations and structure design sub-tasks. Moreover, a multidisciplinary interface model is developed as an extension of the product model to ensure the consistency between the macro- and micro-design processes. The multidisciplinary interface model provides a common representation for the interfaces, which can be used to help the designers to ensure the consistency between the two levels of design process. With the support of the interface, the designers can go back to the general development flow of the macro-level process from the specific design activities, and therefore, the consistency between the two levels of design process is ensured.

**Acknowledgements** This work was in part supported by the Linz Center of Mechatronics (LCM) within the framework of the Austrian COMET-K2 program and the Labex MS2T (supported by the French Government, through the program "Investments for the future" managed by the National Agency for Research—Reference ANR-11-IDEX-0004-02) at the Université de Technologie de Compiègne.

## References

Abramovici M, Bellalouna F (2007) Integration and complexity management within the mechatronics product development. Advances in life cycle engineering for sustainable manufacturing businesses, 14th CIRP Conference on Life Cycle Engineering, Tokyo, Japan, 11–13 June 2007, pp 113–118

Abramovici M, Bellalouna F (2008) Service oriented architecture for the integration of domain-specific PLM systems within the mechatronic product development. 7th international symposium on tools and methods of competitive engineering (TMCE 2008), Izmir, Turkey, pp 941–953

Alvarez Cabrera AA, Foeken MJ, Tekin OA et al (2010) Towards automation of control software: a review of challenges in mechatronic design. Mechatronics 20:876–886

Barbieri G, Fantuzzi C, Borsari R (2014) A model-based design methodology for the development of mechatronic systems. Mechatronics 24:833–843

Bathelt J, Jonsson A, Bacs C, et al (2005) Applying the new VDI Design Guideline 2206 on mechatronic systems controlled by a PLC. International conference on engineering design (ICED 05), Melbourne, Australia. 15–18 August 2005

Blanchard BS (2012) System engineering management. Wiley, New Jersey

Blyler J (2004) Interface management. Instrum Meas Mag 7:32–37

Boehm BW (1981) Software engineering economics. Prentice Hall, New Jersey

Boehm BW (1988) A spiral model of software development and enhancement. Computer 21:61–72

Brezina T, Hadas Z, Vetiska J (2011) Using of co-simulation ADAMS-SIMULINK for development of mechatronic systems. 14th international symposium MECHATRONIKA, Trenín, Slovakia

Bricogne M, Rivest L, Troussier N, Eynard B (2014) Concurrent versioning principles for collaboration: towards PLM for hardware and software data management. Int J Prod Lifecycle Manag 7:17–37

Cao Y, Liu Y, Paredis CJJ (2011) System-level model integration of design and simulation for mechatronic systems based on SysML. Mechatronics 21:1063–1075

Carryer JE, Ohline RM, Kenny TW (2011) Introduction to mechatronic design. Prentice Hall, Boston

Department of Transportation (2007) Systems engineering for intelligent transportation systems. Springer Science+Business Media, Washington

Dorfman M (1990) System and software requirements engineering. IEEE Computer Society Press Tutorial. IEEE CS Press, Los Alamitos, pp 7–22

Fenves S, Foufou S, Bock C, Sriram RD (2006) CPM: a core model for product data. J Comput Inf Sci Eng 8:1–14

Fisher J (1998) Model-based systems engineering: a new paradigm. Incose Insight 1:3–16

Forsberg K, Mooz H (1999) System engineering for faster, cheaper, better. Proceedings of the 9th annual international symposium of the INCOSE1998, Brighton, UK

Fosse E, Delp CL (2013) Systems engineering interfaces: a model based approach. 2013 IEEE aerospace conference, Big Sky, USA

Fotso AB, Wasgint R, Achim R (2012) State of the art for mechatronic design concepts. 8th IEEE/ASME international conference on mechatronic and embedded systems and applications. Suzhou, China. 8–10 July 2012, pp 232–240

Gausemeier J, Moehringer S (2003) New Guideline VDI 2206—A flexible procedure model for the design of mechatronic systems. 14th international conference on engineering design, Stockholm, Sweden, 19–21 August 2003

Gausemeier J, Frank U, Donoth J, Kahl S (2009) Specification technique for the description of self-optimizing mechatronic systems. Res Eng Des 20:201–223

Gausemeier J, Dumitrescu R, Kahl S, Nordsiek D (2011) Integrative development of product and production system for mechatronic products. Robot Comput Integr Manuf 27:772–778

Gautam N, Singh N (2008) Lean product development: maximizing the customer perceived value through design change (redesign). Int J Prod Econ 114:313–332

Hadas Z, Singule V, Vechet S, Ondrusek C (2010) Development of energy harvesting sources for remote applications as mechatronic systems. 14th international power electronics and motion control conference, Ohrid, Macedonia, pp 13–19

Hamraz B, Caldwell NHM, Ridgman TW, Clarkson PJ (2014) FBS Linkage ontology and technique to support engineering change management. Res Eng Des 26:3–35

Hazelrigg GA (1996) Systems engineering: an approach to information-based design. Prentice Hall Upper Saddle River, New Jersey

Hehenberger P (2014) Perspectives on hierarchical modeling in mechatronic design. Adv Eng Inform 28:188–197

Hehenberger P, Poltschak F, Zeman K, Amrhein W (2010) Hierarchical design models in the mechatronic product development process of synchronous machines. Mechatronics 20:864–875

Hoffman D (1990) On criteria for module interfaces. IEEE Trans Softw Eng 16:527–542

Hofmann D, Kopp M, Bertsche B (2010) Development in mechatronics—enhancing reliability by means of a sustainable use of information. 2010 IEEE/ASME international conference on advanced intelligent mechatronics, Montréal, Canada

ISO (1994) Overview and fundamental principles. ISO, Geneva

ISO10303-233 (2012) Industrial automation systems and integration—Product data representation and exchange—Part 233: application protocol: systems engineering. International organization for standardization, Geneva

Jansen S, Welp EG (2005) Model-based design of actuation concepts: a support for domain allocation in mechatronics. International conference on engineering design (ICED 05), Melbourne, Australia, pp 1–15

Kerzhner AA, Paredis CJJ (2012) A SysML-based language for modeling system-level architecture selection decisions. ASME 2012 international design engineering technical conferences and computers and information in engineering conference, Chicago, USA

Kleiner S, Kramer C (2013) Model based design with systems engineering based on RFLP using V6. Proceedings of the 23rd CIRP Design Conference, Bochum, Germany. 11–13 March, pp 93–102

Kolberg E, Reich Y, Levin I (2014) Designing winning robots by careful design of their development process. Res Eng Des 25:157–183

Komoto H, Tomiyama T (2011) Multi-disciplinary system decomposition of complex mechatronics systems. CIRP Ann Manuf Technol 60:191–194

Komoto H, Tomiyama T (2012) A framework for computer-aided conceptual design and its application to system architecting of mechatronics products. Comput Des 44:931–946

Le Duigou J, Bernard A, Perry N (2011) Framework for product lifecycle management integration in small and medium enterprises networks. Comput Aided Des Appl 8:531–544

Lefèvre J, Charles S, Bosch-Mauchand M, et al (2012) Towards multidisciplinary modeling and simulation: Interoperability issues and challenges for mechatronic engineering. Proceedings of TMCE 2012, Karlsruhe, Germany

Li Q, Zhang WJ, Chen L (2001) Design for control-a concurrent engineering approach for mechatronic systems design. IEEE/ASME Trans Mechatron 6:161–169

Lock M (2009) Executive dashboards: the key to unlocking double digit profit growth. Aberdeen, Scotland

Nattermann R, Anderl R (2013) The W-model—using systems engineering for adaptronics. Proc Comput Sci 16:937–946

Noël F, Roucoules L (2008) The PPO design model with respect to digital enterprise technologies among product life cycle. Int J Comput Integr Manuf 21:139–145

Nowak P, Rose B, Saint-Marc L, et al (2004) Towards a design process model enabling the integration of product, process and organisation. 5th international conference on integrated design and manufacturing in mechanical engineering (IDMME'04), Bath, UK, pp 91–103

Object Management Group (2009) Systems modeling language specification. http://www.omg.org/spec/SysML/1.2/PDF/

Pahl G, Beitz W (1988) Engineering design: a systematic approach, 1st edn. The Design Council, London

Pandikow A, Herzog E, Törne A (2000) Integrating systems and software engineering concepts in AP-233. Proceedings of the 2000 INCOSE Symposium, INCOSE, pp 831–837

Penas O, Plateaux R, Choley J-Y et al (2011) Conception mécatronique—vers un processus continu de conception mécatronique intégrée. Techniques de l'ingénieur BM 8(020):1–23

Pratt MJ (2001) Introduction to ISO 10303—the STEP standard for product data exchange. J Comput Inf Sci Eng 1:102–103

Qamar A, Paredis CJJ (2012) Dependency modelling and model management in mechatronic. Proceedings of the ASME 2012 international design engineering technical conferences and computers and information in engineering conference (IDETC/CIE 2012), Chicago, USA

Sellgren U, Törngren M, Malvius D, Biehl M (2009) PLM for Mechatronics integration. International conference on product lifecycle management, Bath, UK, 6–8 July 2009

Seyff N, Maiden N, Karlsen K et al (2009) Exploring how to use scenarios to discover requirements. Requir Eng 14:91–111

Shetty D, Kolk RA (2010) Mechatronics system design, SI version. Cengage Learning, Boston

Stone R, Wood K (2000) Development of a functional basis for design. J Mech Des 122:359–370

Tomiyama T, Gu P, Jin Y et al (2009) Design methodologies: industrial and educational applications. CIRP Ann Manuf Technol 58:543–565

Tomizuka M (2002) Mechatronics: from the 20th to 21st century. Control Eng Pract 10:877–886

Torry-Smith JM, Mortensen NH, Achiche S (2013) A proposal for a classification of product-related dependencies in development of mechatronic products. Res Eng Des 25:53–74

Umeda Y, Takeda H, Tomiyama T, Yoshikawa H (1990) Function, behaviour, and structure. Applications of artificial intelligence in engineering V. Computational Mechanics Publications, Springer-Verlag, Berlin, pp 177–193

Umeda Y, Ishii M, Yoshioka M et al (1996) Supporting conceptual design based on the function-behavior-state modeler. Artif Intell Eng Des Anal Manuf 10:275–288

Van Beek TJ, Erden MS, Tomiyama T (2010) Modular design of mechatronic systems with function modeling. Mechatronics 20:850–863

Vasić VS, Lazarević MP (2008) Standard industrial guideline for mechatronic product design. FME Trans 36:103–108

VDI 2206 (2003) Design handbook 2206. Design methodology for mechatronic systems. VDI Publishing Group, Düsseldorf

Weber C (2005) CPM/PDD—An extended theoretical approach to modelling products and product development processes. 2nd German Israeli symposium on advances in methods and systems for development of products and processes, Stuttgart, Germany

Weber C (2014) Modelling products and product development based on characteristics and properties. An anthology of theories and models of design. Springer, London, pp 327–352

Zha XF, Fenves SJ, Sriram RD (2005) A feature-based approach to embedded system hardware and software co-design. ASME design engineering technical conference, Long Beach

Zheng C, Bricogne M, Le Duigou J, Eynard B (2014) Survey on mechatronic engineering: a focus on design methods and product models. Adv Eng Inform 28:241–257

Zheng C, Le Duigou J, Bricogne M, Eynard B (2015a) Multidisciplinary interface modelling: a case study on the design of 3D measurement system. 12th international conference on product lifecycle management, Doha, Qatar

Zheng C, Le Duigou J, Bricogne M, Eynard B (2015b) Design process for complex systems engineering based on interface model. Incose Insight 18:22–24

Zheng C, Le Duigou J, Bricogne M, Eynard B (2016) Multidisciplinary interface model for design of mechatronic systems. Comput Ind 76:24–37