INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED11
15 - 18 AUGUST 2011, TECHNICAL UNIVERSITY OF DENMARK

INTERDISCIPLINARY SYSTEM MODEL FOR AGENT-SUPPORTED MECHATRONIC DESIGN

Ralf Stetter[1], Holger Seemüller[1], Mohammad Chami[1] and Holger Voos[2]
(1) Hochschule Ravensburg-Weingarten, D (2) University of Luxembourg

ABSTRACT

This paper presents research results from an ongoing project concerning the development of agent systems which aim to support mechatronic design. Today mechatronic design usually suffers from a lack of domain-spanning IT-support. The abundance of logical connections between the disciplines is only present in the designer’s minds or in unstructured documents. One approach to document and use the connections between the disciplines are agent based systems. Such systems use independent software entities representing either components of the product or certain process segments which interact in a system called agent system. Due to their flexibility and the ability to achieve solutions which satisfy multiple objectives such systems are promising approaches to address the complexity challenge of mechatronic design. However, the application of such systems requires the documentation of the interdisciplinary connections in an interdisciplinary system model. In this paper SysML is proposed for this specific task. This proposal is supported by the application to an exemplary product development – the parametric development of a Quadrotor – an unmanned multi-purpose flying object.

Keywords: Mechatronic design, Agent Based Systems, SysML, Interdisciplinary System

1 INTRODUCTION

“Mechatronic Design” describes the synergetic creation and integration of mechanical engineering, electrical engineering and information technology for the specification and description of any kind of physical products and processes [1]. Today mechatronic design usually suffers from a lack of domain-spanning IT-support. Each discipline is using its specific IT-tools (e. g. CAD systems) and data formats and the interdependencies between the disciplines are not documented in a consistent manner (frequently unstructured documents such as MS Word or MS PowerPoint files are used). This approach has significant drawbacks on the consistency of interdisciplinary model data and affects the efficiency of the overall mechatronic design process. Many approaches to unify the tools and data during the last two decades were made but have not made their way into the product development departments in industrial practice.

From research and industry the idea of an interdisciplinary system model holding cross-domain information about the system and important relationships has been arisen [2], [3]. So the unstructured documents and implicit knowledge of the designers can be summarized and captured in a structured way for later reuse.

However a gap between this interdisciplinary system model and the respective IT-tools of the involved domains still remains. The research presented in this paper proposes an approach to minimize this gap by using a multi-agent system to manage the system model which is based on SysML. In that way the coordination and data consistency between the involved designers and engineers shall be improved and the overall mechatronic design process shall be enhanced.

The remainder of this paper is structured as follows. Section 2 explains the context of the work and presents the needed background information. Section 3 shows the overall concepts of our approach where Section 4 explains them with the help of an example. Section 5 lists possible use cases of the approach. Section 6 concludes the paper and gives an outlook.

2 EXISTING APPROACHES AND BACKGROUND WORK

The following section serves as basis for the further sections through explaining important characteristics of interdisciplinary engineering, summarizing the basics of SysML and giving on overview of agent based systems.

2.1 Interdisciplinary Engineering

Due to the domain-spanning characteristics of mechatronic systems, interdisciplinary approaches and methods are needed to address the needs of all involved engineering domains equally together with their strong interrelationships even in early phases of the mechatronic design process. Although this idea is not new in research and has been frequently discussed, e. g. in [2], [3] and [4], it has been hardly realized due to the lack of tools supporting collaboration across disciplines.

From the engineering process perspective, the VDI 2206 guideline [2] gives a methodological support for the cross-domain development of mechatronic systems. The guideline is mainly based on the V model used commonly in software development which is adapted to the special needs of mechatronic product design. One primary idea is the creation of an interdisciplinary principal solution during the system design phase which describes the system in a domain independent way. For the domain-specific design phase this solution shall be partitioned into the respective domains which then design their parts concurrently. However no method or tool is proposed for the creation of the principal solution.

Chen et al. [3] propose a constraint modeling-based approach by modeling the components of mechatronic systems as objects with attributes, and by identifying and modeling the constraints between these attributes.

From an organizational perspective Pahl et al. describe in [4] the interdisciplinary cooperation for the organizational and operational structures and focus on the prerequisites for forming interdisciplinary teams by structuring the system in such a way that its properties can be modeled precisely and by defining the interfaces between the process steps more precisely and ambiguously.

Currently, a need for further research is still apparent, because many approaches from academia have not found their way into industry yet. In this paper, we see the handling of the interdisciplinary relationships between the different domain specific models as an important aspect to enhance interdisciplinary product engineering. These relationships have to be captured and documented on one side but also be made usable for engineers on the other side.

2.2 SysML

The System Modeling Language (SysML) is “a general-purpose graphical modeling language for specifying, analyzing, designing and verifying complex systems that may include hardware, software, information, personnel, procedures and facilities” [5]. SysML supports the model-based systems engineering approach by offering modelers the ability to create an overall system model of an interdisciplinary system. This overall system model covers different aspects like requirements, parametric, structure or behavior together with their interrelationships. SysML represents a subset of UML2 with an extension needed to satisfy the requirements of the UML for Systems Engineering Request for Proposal (UML for SE RFP) that was initiated in 2003 by OMG [5].

SysML, as any other modeling language, needs a methodology to collect, analyze, and model the different aspects of a system. Domain experts have already described several methodologies for modeling with SysML. Weilkiens introduced SYSMOD in [6] as a pragmatic approach that uses SysML to model systems from analysis to design. A computational product model in [7] shows a way to use SysML for the conceptual design phase according to VDI2221 [8]. In [9], an approach is presented that uses SysML to create discipline neutral system-level models for complex mechatronic systems. A model integration framework is demonstrated in [10] by using SysML profiles to model different discipline specific views in SysML which can be transformed into respective domain-specific models. Thramboulidis proposed in [11] a synergistic integration of the constituent parts of mechatronic systems by using SysML to specify the central view-model of the mechatronic system.

However, despite the effective role that SysML plays in holding the important interdisciplinary relationships, there is still a lack of usage of these relationships in current industrial practice (according to several discussions with engineers and managers).

2.3 Agent-based Systems

Multi-agent-systems which are based on intelligent software agents have been the object of research since the middle of the nineties of the last century. The core is composed of autonomous software components which interact in a goal directed manner and communicate with each other and other components in the scope of a multi-agent system. Using this communication and appropriate algorithms, these software agents are able to coordinate their behavior and to solve problems together without any central control. The main advantage of this decentral approach (decentral in the sense that no centralized control exists) is its large flexibility. Commonly the result (the solution of the problem) is not an optimum result in the sense of an optimization calculation but a result which fulfills all the complex boundary conditions of such a system. A profound overview of the state of the art can be found in Weiss [12], Luck&d'Inverno [13], Trencansky&Cervenka [14] and Padgham et al. [15]. Different software architectures for multi-agent systems are available [12]. The BDI concept describes an architecture where agents are described and developed in terms of their beliefs, desires and intentions [16].

Intelligent software agents are already applied in pilot plants for the planning and control of production systems [17], [18], [19], [20], [21]. In this application agents are the representatives of production machines as well as of parts to be produced. These agents then start negotiations, in order to achieve an appropriate assignment of production orders to production machines. Further areas of application for intelligent software agents can also be found in defense and robotics applications [16].

The application of intelligent software agents in engineering processes is investigated in first research works by Huang et al. [22], Tseng et al. [23], Mild&Taudes [24] and Zhang et al. [25]; the main emphasis is on web-based collaborative work. No special emphasis is given to supporting the development of mechatronic products; furthermore the inclusion of elaborate engineering tools as well as the process planning and control are only covered in a rudimentary manner.

3 AGENT-BASED ENGINEERING OF MECHATRONIC PRODUCTS WITH SYSML

This approach shall combine the advantages of interdisciplinary system modeling with SysML together with the principles of agent based systems. The basic idea is an agent-based management of a SysML model. Based on the modeled information about the mechatronic system the agents shall negotiate and handle the interdisciplinary dependencies across the system during the development process [26]. Therefore the SysML model has to be transformed into single agents which posses the knowledge previously modeled in SysML (Figure 1 – upper left).

The purpose of using a multi-agent system lies in the concept that each component of the system can be represented by an autonomous agent who takes care about relationships to other agents. In case of a conflict the involved agents can negotiate and forward new information to the respective engineer. In that way data consistency between the involved domains is guaranteed and the interrelationships between discipline specific model data can be considered.

For the realization of this idea, the SysML model has to be created in a form that the multi-agent system can deal with. Therefore the semantics of the SysML model has to be defined clearly. The following sections describe the theoretical foundations we base our approach on, the semantics of the SysML model according to the presented foundation as well as the way the model is transformed into a multi-agent system.

Additionally the multi-agent system can basically be used for engineering design process planning and enactment (e. g. based on SPEM compare section 5). Here an agent represents an engineering activity and uses product information from the system model to guide engineers through the process. Figure 1 presents an overview about the architecture of the multi agent system. However, the focus of this paper is the management of the system model with agents (left side in Figure 1).

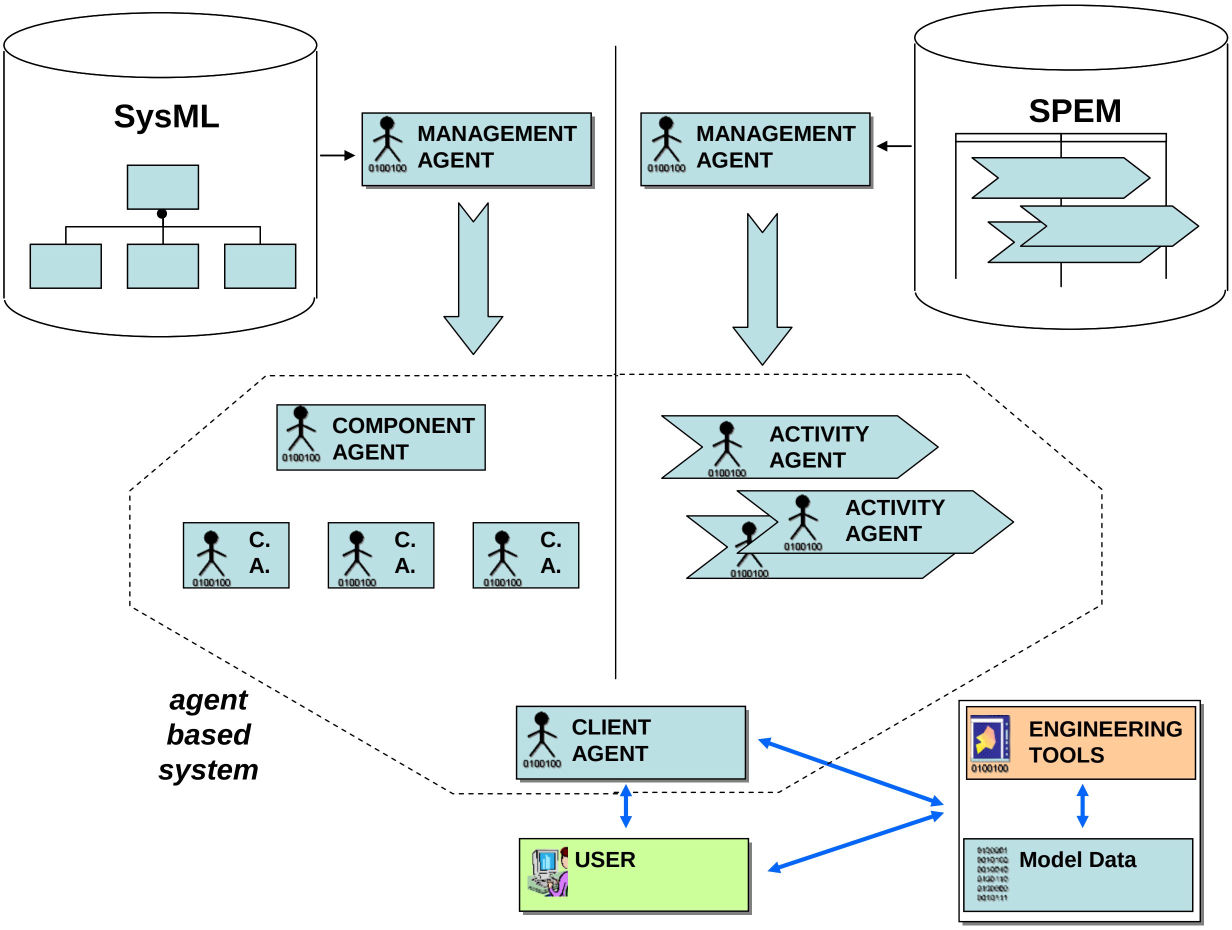


Figure 1. Agents based systems architecture

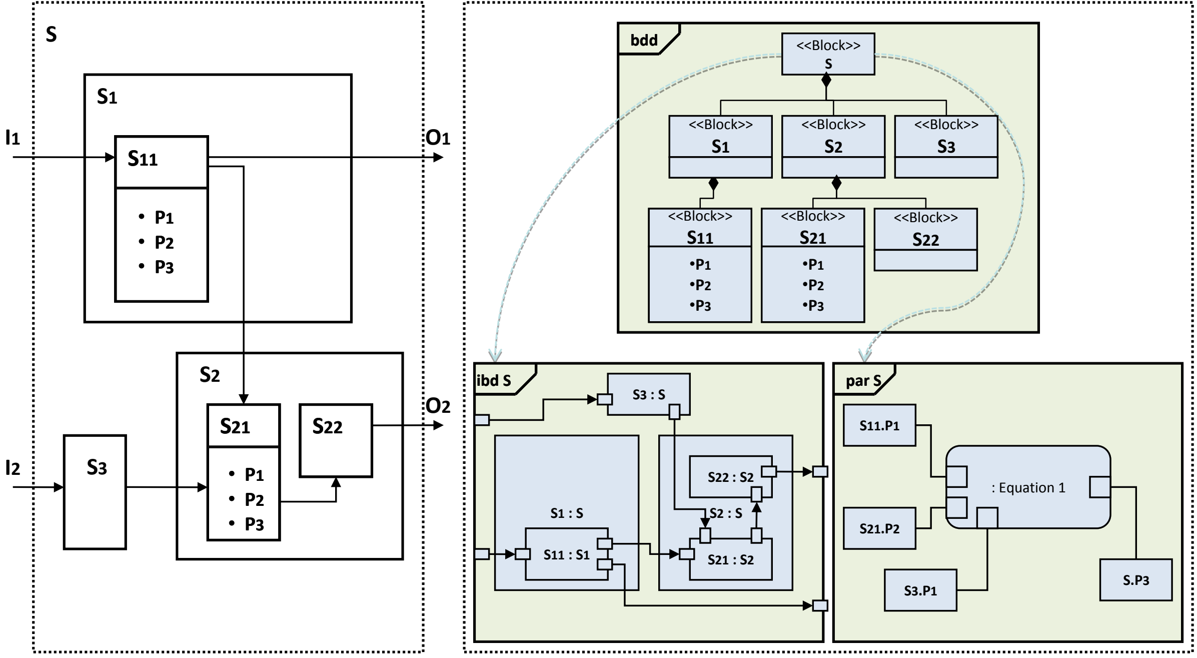
3.1 Theoretical Model of Mechatronic Systems

In order to deal with a mechatronic system the composition of such systems has to be defined first. According to Stetter&Voos [27] a mechatronic system is assumed to consist of single components connected with interfaces. These components do not necessarily have to belong to a single discipline. So for the design of an electrical motor, a CAD engineer may be responsible for the geometry while a control engineer is modeling the behavior of the device with differential equations. While both engineers are working concurrently on their domain specific model, strong interdependencies such as the dimension of an axis exist within the single interdisciplinary component.

Furthermore it is assumed that an overall system can be hierarchically decomposed into system, subsystem and component level whereas the semantics of these terms are project dependent and cannot be defined in general. These compositional elements can be connected via structural interfaces as well as material, energy and information flow relationships [2].

Additionally interdisciplinary relationships as they can appear within a single component may also exist between distinctive components. So, if the maximum force of the electrical motor changes due to an adoption of the electrical model of this component also the source code of the motor controller has to be adapted.

In order to deal with these aspects, a theoretical model is needed that examines the fundamentals of the system. A simplified model of a mechatronic system is shown in left side of Figure 2 and is described more in detail from a mechanical construction point of view in [4].



System-Model: **SysML-Model:**

S: overall System, **Si**: subsystem,
Sij: subsystem of **Si**, **Pi**: property
Ii: input,**Oi**: output

bdd: SysML Block Defintion Diagram
ibd: SysML Internal Block Diagram
par: SysML Parametric Diagram

Figure 2. Theoretical Model

The contribution of this paper deals with the following fundamental aspects of an interdisciplinary system:

- the hierarchy of interdisciplinary components together with their respective parameters and
- the interrelationships between components: structural relationships (interfaces) and flow relationships (material flow, energy flow and/or information flow) as well as
- the interrelationships between disciplines (intra-component relationships or inter-component relationships).

3.2 SysML System Model

As mentioned in section 2.2, a methodology is required for SysML to guide the modeler, with a significant freedom, during the modeling process. In this approach the block definition diagram, the internal block diagram and the parametric diagram from SysML are being used with a given semantics to model a mechatronic system. It is assumed that during a conceptual design phase, a cross-domain system model has been developed according to the assumptions presented in section 3.1. This section describes how this theoretical system model can be realized with SysML for further use with the multi-agent system.

The block definition diagram serves as an appropriate means to model the decomposition of the overall system. Here, a SysML block is used to represent a mechatronic system, subsystem or component as it can be used to represent any type of “thing” that exists in the real world [28]. The composition association expresses the hierarchical relationships between system, subsystem and component layer. The right part of Figure 2 shows an example of a block definition diagram representing the system breakdown structure of the theoretical system of section 3.1.

The interrelationships between components can be modeled with the internal block diagram. Therefore, SysML offers adequate means to express the intended interrelationships between components by using the port concepts which exists of standard ports and flow ports [5] to express different kind of interfaces. The structural relationships will be modeled with a binding connector between two standard ports. Flow relationships including a flow of material, energy or information between blocks can be modeled with an itemflow between flow ports.

For modeling interrelationships between disciplines the parametric diagram serves as an ideal basis as properties of blocks can be related with dependencies as well as constraints using the constraint block from SysML. Constraint blocks offer the possibility to express relationships in e. g. a mathematical equation. In our approach two different kinds of relationships have to be considered: a mathematical relationship as well as a pure dependency relationship of information which can appear between two properties as well as between a property and a block. The latter can be easily expressed by the usage of the dependency relationship. The former relationship will be modeled by using a constraint block and directly entering the mathematical relationship.

Currently, one of the most benefits of SysML is its domain independent syntax. Consequently it can serve the role as a common language between different designer's knowledge. More specifically, in the way of modeling, two aspects of knowledge propagation have to be considered: A top-down way from system modelers and a bottom-up way from domain specific modelers. Different types of knowledge are captured and hold in one modeling language, knowledge sources can be expertise, system level designers, and other various stakeholders. This idea improves directly the issues of knowledge-sharing, traceability, navigation and the transition between the domain independent design phase and the domain specific phase of the V-model [26].

3.3 Transformation

To enable an agent-based management of the interdisciplinary system (SysML) model it is transferred into an executable multi-agent system, i. e. information about product characteristics and structures are transferred to the agent based system. This transformation is reached via a central management agent who is capable to read an existing model stored in XMI format [29]. The management agent extracts necessary information and creates the respective agents from this information. Additional model data which is not necessary for creating the agents will not be considered. In that way modelers are still free to add extra model data to the system model which may facilitate the expressiveness of the overall system model.

The management agent is running on a central server where the modeler can upload the current model. Triggered by an upload the management agent analyses the model and updates the agents.

- The transformation rules are straightforward:
  1. Each block which is not part of another block is considered to be at the highest level of the system hierarchy and therefore transformed into an agent.
  2. Beginning from point 1 all part associations are transformed into single agents
  3. The parts of parts are also transformed into single agents
  4. Step 3 is repeated recursively until the leafs of the hierarchy tree is reached
- The beliefbase of any single agent contains the following information: the properties of the block, the parts properties of the block, the interfaces of the block, the flow relationships and the modeled relationships between discipline-specific parameters

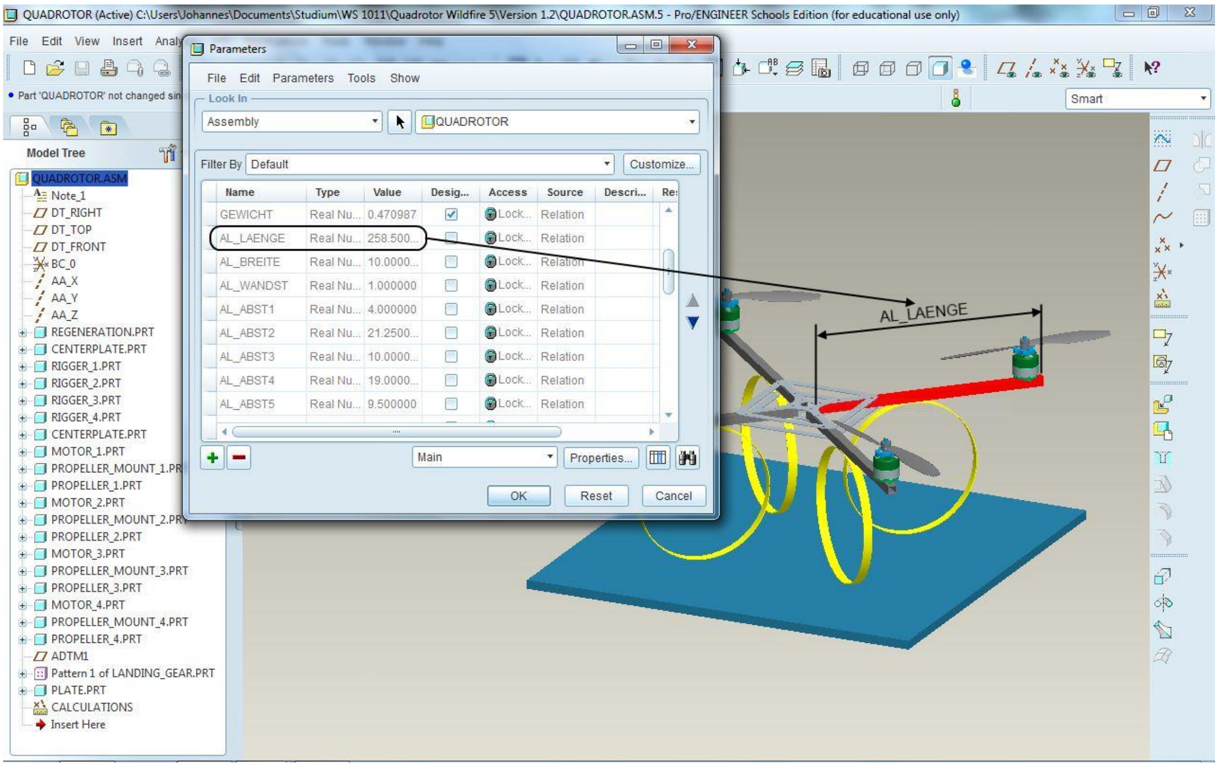
So a net of SysML agents is created representing the components of the overall system and which is running on a central server and can be accessed by any engineer within the network.

To work with these component agents, client agents are running on the workstations of the respective engineer to serve as an interface between the engineer who is directly working with his domain-specific model data and the multi-agent system. The purpose of the client agent is the negotiation with the component agents when important changes within his field of responsibility occur. Each transformed component agent has the main goal to observe and take care on the modeled properties. If a client agent notifies a component agent about a change of one of its properties, the component agent will check its beliefbase for existing interrelationships between this property and others within the same component or others. In case of a relationship it will inform the respective component agent about the change. This agent now has the responsibility to inform the respective engineer via the associated client agent.

Obviously a danger is present that such systems may explode beyond usage of human clients when the complexity of the product increases. However, the complexity of all kinds of mechatronic products will most probably further increase in the coming years and decades. Approaches which realize a part of the connection between domains and negotiation of complex product characteristics without human interference by means of intelligent systems are therefore promising. In future such approaches have to be combined with approaches which can reduce the complexity and create certain views for human clients.

4 EXAMPLE

The UAV (unmanned aerial vehicle) Quadrotor project, running at the Hochschule Ravensburg-Weingarten in Weingarten, Germany, is currently in the design stage of a rotor-craft equipped with four powered rotors laid up symmetrically around its center. This kind of unmanned multi-purpose flying object is characterized by its good controllability and may find its application in surveillance operations of traffic or sports events. The Quadrotor is considered as a mechatronic system with rather low complexity. It is a typical example of a mechatronic product consisting of distinct mechanical, electrical and IT-based elements. Different students with different backgrounds are involved in the development process and directly supervised by the professor in charge. After having planned the project, a list of concrete requirements and functional analysis was achieved. From these requirements different tasks were assigned to the students in order to be realized in a given period of time. A group of students took the structural body design aspect of the Quadrotor; a detailed CAD-model of the Quadrotor was generated using ProEngineer. This CAD-model was designed in a completely parameterized manner. It is now possible to change one major dimension (e. g. the length of one of the main brackets) and all other components will change so that the general performance parameters of the Quadrotor will still be achieved. This logic is right now realized in ProEngineer by means of Layout Parameters (Figure 3) and external logic in Microsoft Excel.


QUADROTOR (Active) C:\Users\Johannes\Documents\Studium\WS 1011\Quadrotor Wildfire 5\Version 1.2\QUADROTOR.ASM.5 - Pro/ENGINEER Schools Edition (for educational use only)

Part 'QUADROTOR' not changed sin

Parameters

Look In: Assembly | QUADROTOR

Filter By: Default

Name	Type	Value	Desig...	Access	Source	Descri...	Re
GEWICHT	Real Nu...	0.470987	☑	Lock...	Relation		
AL_LAENGE	Real Nu...	258.500...	☐	Lock...	Relation		
AL_BREITE	Real Nu...	10.0000...	☐	Lock...	Relation		
AL_WANDST	Real Nu...	1.000000	☐	Lock...	Relation		
AL_ABST1	Real Nu...	4.000000	☐	Lock...	Relation		
AL_ABST2	Real Nu...	21.2500...	☐	Lock...	Relation		
AL_ABST3	Real Nu...	10.0000...	☐	Lock...	Relation		
AL_ABST4	Real Nu...	19.0000...	☐	Lock...	Relation		
AL_ABST5	Real Nu...	9.500000	☐	Lock...	Relation		

Figure 3. Quadrotor Design in ProEngineer

In this parametric model it is possible to change parameters (e. g. the parameter AL_LAENGE in figure 3) in one of the IT-systems (ProEngineer or Excel) and the other parameters will be automatically adapted and even other motors my be chosen from a certain selection. Through this always a Quadrotor design will result which is able to lift its own weight and a given additional weight.

This logic of the parametric model can in future be used by an agent system in order to allow the usage of this logic also in other domains. (compare also section 4.3 - Figure 6 shows an example of the current parametric logic depiction).

The Quadrotor project is used to evaluate the approach described in this paper. A SysML model for the Quadrotor is achieved to represent the breakdown structure of the Quadrotor and to describe the existing interdisciplinary relationships between its components. The aim of this model is to enhance the collaboration between the different disciplines for a better integration on different levels. In the following, the modeling mechanism of the Quadrotor in SysML is described more in details.

4.1 The Quadrotor Block Definition Diagram:

Figure 2 shows the block definition diagram of the Quadrotor, where the breakdown structure is modeled. This diagram can be read as follows: the Quadrotor project is made up of three interacted systems: the operator, the Quadrotor and the environment where the Quadrotor is suppose to fly. The Quadrotor, as a system, is composed of several mechatronic components. Some components such as the motor are expressed with one block but with several composition associations. Furthermore, other components are again decomposed into their parts, as shown in Figure 4, till the necessary level of decomposition is acknowledged. Each element of the Quadrotor is modeled with a block and each block contains its attributes, ports and interfaces with other blocks. Figure 4 on the right shows another view of the block motor, its specifications are listed in the attributes compartment and two flow ports represent the input/output interfaces with its environment.

The transformation algorithm reads this diagram and creates for every component a single agent. Concretely, beginning from the root of the break down structure each part association represented with a black diamond will be transformed into a respective agent. In that way, e.g. one agent representing the camera and four agents representing each motor will be created. Additionally, the properties and modeled ports will be stored in the beliefs of the respective agent.

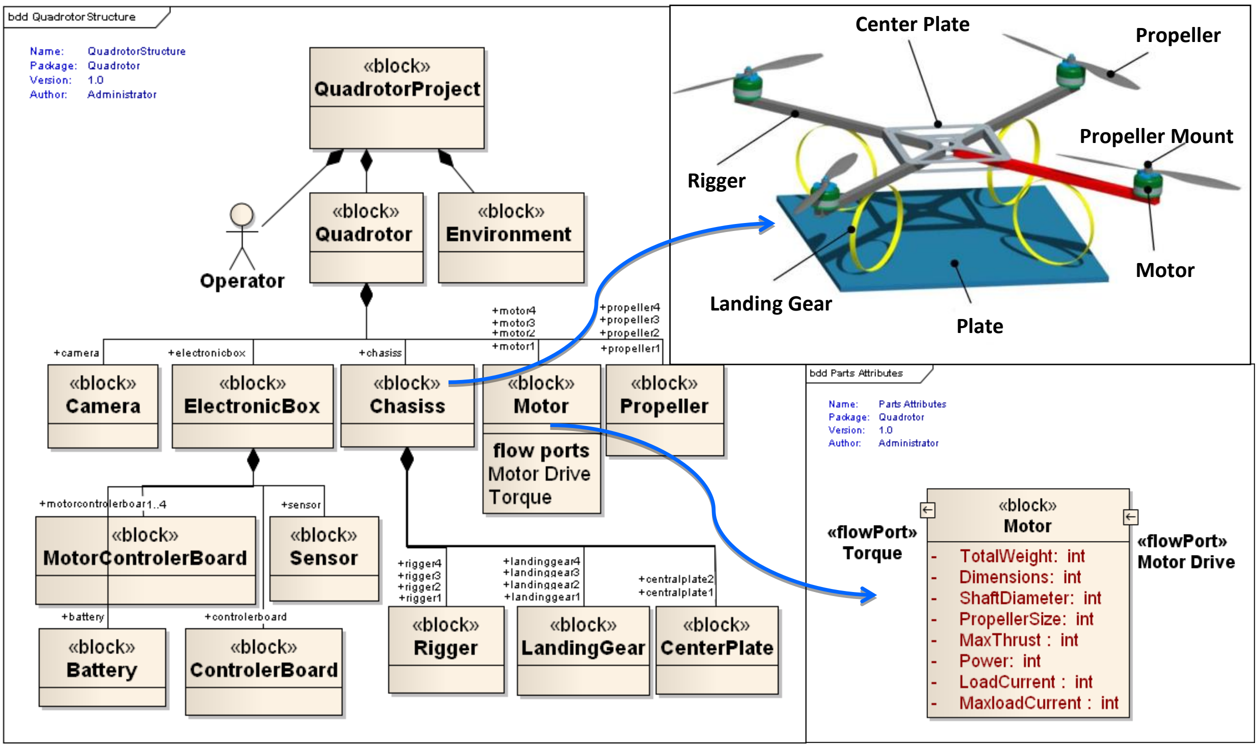


Figure 4. The SysML block definition diagram of the Quadrotor project

4.2 The Quadrotor Internal Block Diagram

An internal block definition diagram can be created to represent the internal structure of this block and the interfaces between its parts. Figure 5 shows an internal block definition diagram of the block Quadrotor where the internal structure is modeled and the interrelationships between its components are represented. One of the block properties of SysML, the part property, is shown in the internal block definition diagram with a solid-outlined box [5] and is used to represent a part of an existing block from the block definition diagram.

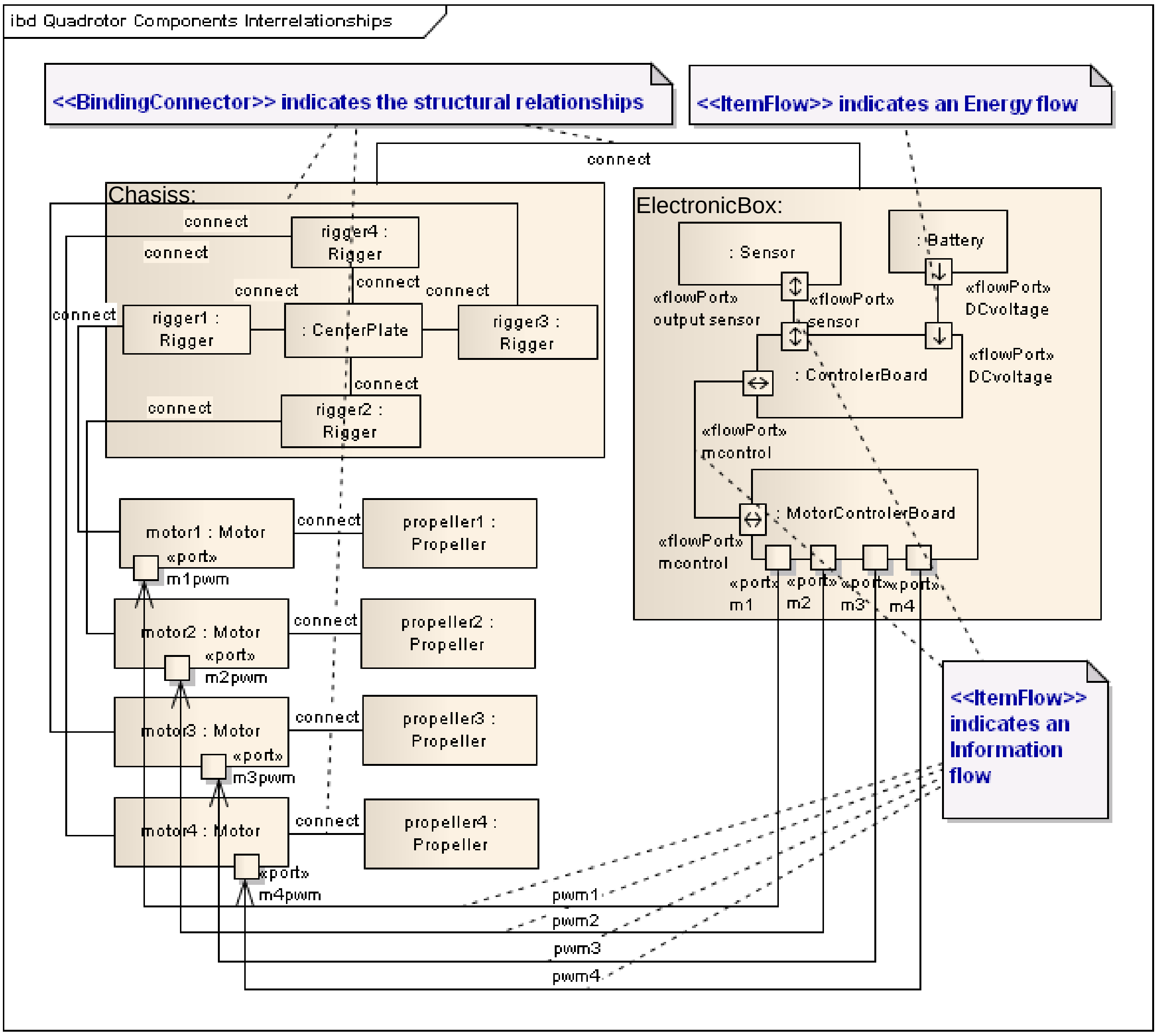


Figure 5. SysML internal block diagram of the Quadrotor

Structural relationships indicating hardware interfaces between parts, such as an assembly connection of a motor with its respective rigger and its propeller is modeled with a binding connector. An information flow of the DC-motor speed control, the PWM signal, between the motor controller board and the motor is modeled with a SysML itemflow. An energy flow can be seen between the battery and the main controller board.

In this way, the more interrelationships information is added to the SysML model with the internal block diagram, the greater the relationships-beliefbase between the agents is achieved.

4.3 The Quadrotor Parametric Diagram

During the domain-specific design of the Quadrotor, engineering students from different disciplines were dealing with their own specialized set of tools to design, simulate or analyze. As described in previous work [26], there is a need to increase the interoperability between existing tools and an all-in-one development suite tool for mechatronic system would not be affordable and even not accepted by industry. We see by using the parametric diagram of SysML an approved approach to represent the interrelationships between the disciplines and later transfer it to the agents' beliefbase. It is important to notice that the parametric diagram is not replacing any of the existing tools, instead it is linking between them by modeling the interdependencies between the properties of the components each tool have to deal with. On the left side of Figure 6 an inter-component relationship diagram is modeled to calculate the total weight of the Quadrotor (as a critical aspect during the development). This diagram holds the weight properties of the components from a wide range of tools. On the right side of Figure 6, an intra-component relationship is modeled to integrate between two disciplines, one from a control engineer student working with Matlab/Simulink to simulate the control algorithms for flying the

Quadrotor and other one from an electronic engineer student dealing with the design and software implementation for the motors.

The information coming from the parametric diagram is used during the agent transformation to add the knowledge about the relationships between the disciplines to the agents.

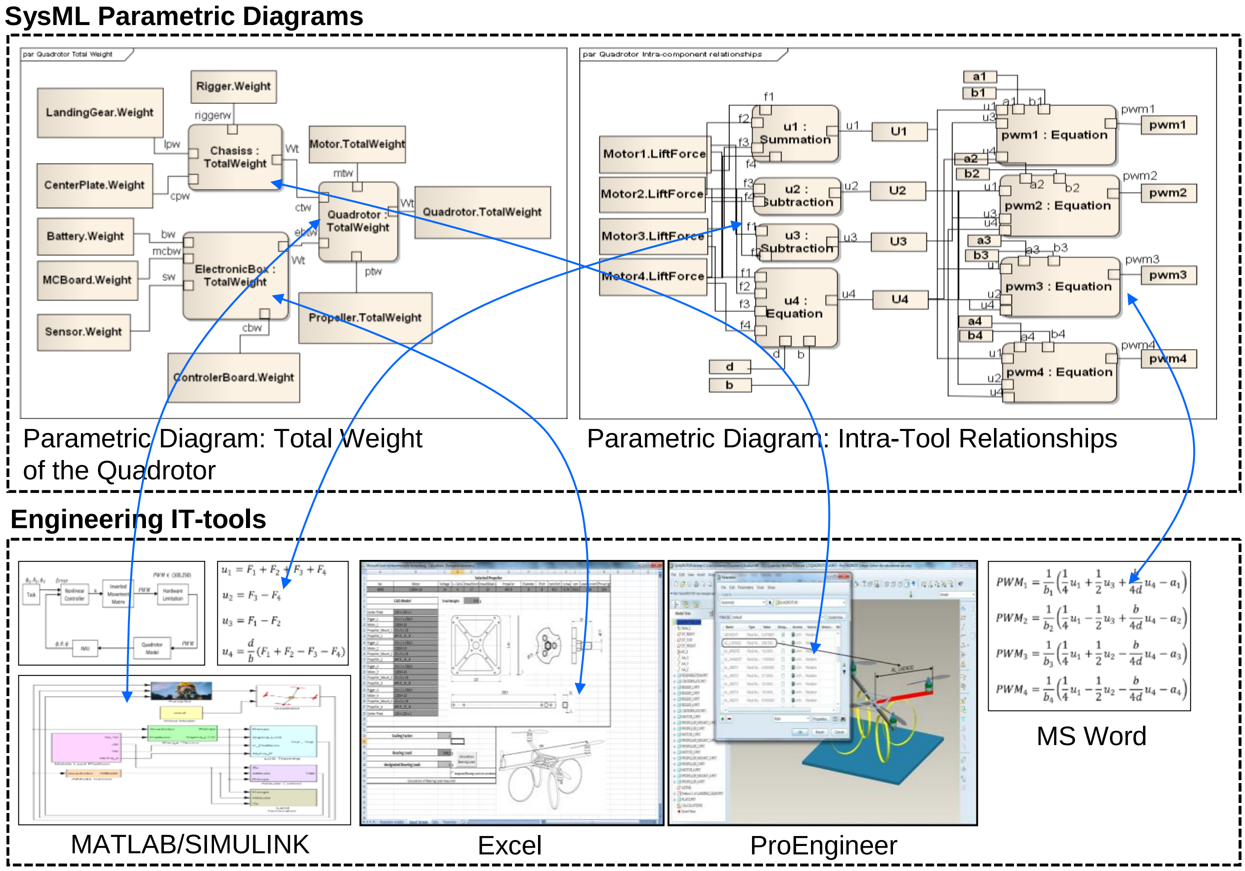

SysML Parametric Diagrams
par Quadrotor Total Weight
LandingGear.Weight
Rigger.Weight
Chassis : TotalWeight
Motor.TotalWeight
CenterPlate.Weight
Quadrotor : TotalWeight
Quadrotor.TotalWeight
Battery.Weight
MCBoard.Weight
ElectronicBox : TotalWeight
Sensor.Weight
Propeller.TotalWeight
ControlerBoard.Weight
par Quadrotor Intra-component relationships
Motor1.LiftForce
Motor2.LiftForce
Motor3.LiftForce
Motor4.LiftForce
u1 : Summation
u2 : Subtraction
u3 : Subtraction
u4 : Equation
U1
U2
U3
U4
pwm1 : Equation
pwm2 : Equation
pwm3 : Equation
pwm4 : Equation
pwm1
pwm2
pwm3
pwm4
Parametric Diagram: Total Weight of the Quadrotor
Parametric Diagram: Intra-Tool Relationships
Engineering IT-tools
$u_1 = F_1 + F_2 + F_3 + F_4$
$u_2 = F_3 - F_4$
$u_3 = F_1 - F_2$
$u_4 = \frac{d}{b}(F_1 + F_2 - F_3 - F_4)$
$PWM_1 = \frac{1}{b_1}\left(\frac{1}{4}u_1 + \frac{1}{2}u_3 + \frac{b}{4d}u_4 - a_1\right)$
$PWM_2 = \frac{1}{b_2}\left(\frac{1}{4}u_1 - \frac{1}{2}u_3 + \frac{b}{4d}u_4 - a_2\right)$
$PWM_3 = \frac{1}{b_3}\left(\frac{1}{4}u_1 + \frac{1}{2}u_2 - \frac{b}{4d}u_4 - a_3\right)$
$PWM_4 = \frac{1}{b_4}\left(\frac{1}{4}u_1 - \frac{1}{2}u_2 - \frac{b}{4d}u_4 - a_4\right)$
MATLAB/SIMULINK
Excel
ProEngineer
MS Word

Figure 6. SysML parametric diagrams of the Quadrotor with different it-tools

5 PROCESS INTEGRATION

The multi-agent system helps engineers during the domain-specific design phase of the overall mechatronics design process. However a pure representation of the system model with agents is not sufficient for a successful mechatronics design process as only product information is stored in the model. Also, for a convenient application of the system, process information has to be integrated in a way that domain-specific activities can be allocated and enacted. This information is modeled with SPEM [30], a modeling language for software engineering processes which has the potential to fit to the needs of mechatronics design activities.

The right part of Figure 1 shows the idea of activity agents representing engineering activities. Together with the client agents which serve as an interface to the engineers, relationships between these three types of agents are established to enact the design process. An activity agent can belong to one or more component agent while one component agent can also be related to several activity agents. Additionally client agents are assigned to one or more activity agents. With these relationships, discipline-specific engineers can be instructed to perform activities which are related to modeled system elements within their domain, e.g. CAD for mechanics.

As mechatronic components involve diverse domains, the activities have to be aligned carefully. It is obvious that interdisciplinary relationships have to be considered. The activity agents can negotiate with the component agents about their interrelationships and then align themselves accordingly into the right order. Here three different use cases have been identified in which the multi-agent system can facilitate the cooperation.

1. Parallel and independent activities can be executed concurrently without any interrelation in between.

2. Completely dependent activities have to be aligned in a sequential order as an activity may need the result of another activity.
3. Partial dependent activities are the most interesting case for this approach as the multi-agent system can help managing the dependencies in between these process steps. As the relationships are already known by the system model the respective activities may start concurrently with the knowledge that one activity needs some information from another activity. Triggered by the occurrence of this needed information (e.g. a parameter value) the multi-agent system can automatically inform the requesting activity. Furthermore in the case of a change of this information during the following steps, the former requesting activity can get informed. In that way concurrent engineering is facilitated and the cooperation between the disciplines is enhanced.

6 OUTLOOK

This paper presented an approach to enhance interdisciplinary system modeling for mechatronic with SysML by using the model with a multi-agent system during domain-specific activities. In that way data consistency is warranted. Example mechatronic design processes show the general feasibility of this innovative approach. Obviously, a number of issues still have to be addressed and elaborate software prototypes have to be realized.

Future work will deal intensively with the integration of process information into the existing multi-agent system which currently just represents product information. In detail the transformation of the process description modeled in SPEM notation into distinctive agents and the way to establish the relationships between the component agents, the activity agents and the client agents have to be analyzed and implemented into the existing prototype.

The usage of agents for requirement validation is also an interesting facet of the approach as SysML offers the modeling of requirements and will be validated.

REFERENCES

[1] Möhringer S. and Stetter R. A Research Framework for Mechatronic Design. In: Proceedings *of the 11th International Design Conference DESIGN 2010*, Dubrovnik – Croatia, 2010.

[2] VDI 2206. *Design methodology for mechatronic systems*. Beuth, Berlin, 2004.

[3] Chen K., Bankston J., Panchal J. H. and Schaefer D. *A Framework for the Integrated Design of Mechatronic Systems, in Collaborative Design and Planning for Digital Manufacturing*, (L. Wang and A. Nee, Editors), Springer, 2009. pp. 37-70.

[4] Pahl G., Beitz W., Feldhusen J., and Grote K.-H. *Engineering design: A systematic approach, (3rd ed.),* Springer-Verlag, 2007.

[5] Object Management Group. "OMG Systems Modeling Language (OMG SysML^{TM})," available at http://www.omgsysml.org, November 2008.

[6] Weilkiens T. *Systems Engineering with SysML/UML*. Elsevier, Ed. Morgan Kaufmann Publishers Inc, June 2008.

[7] Wölkl S. and Shea K. A Computational Product Model for Conceptual Design Using SysML. In *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, San Diego, 2009.

[8] VDI Design Handbook 2221. *Systematic Approach to the Design of Technical Systems and Products*. Düsseldorf: VDI-Verlag 1987.

[9] Follmer M., Hehenberger P., Punz S. and Zeman, K . Using SysML in the Product Development Process of Mechatronic Systems. In *Proceedings of the 11th International Design Conference DESIGN 2010*.

[10] Aditya A. Shah, Schaefer D and Christiaan J.J. Paredis. Enabling Multi-View Modeling With SysML Profiles and Model Transformations. *International Conference on Product Lifecycle Management 2009*.

[11] Thramboulidis K. The 3+1 SysML View-Model in Model Integrated Mechatronics. *Journal of Software Engineering and Applications (JSEA), 2010*.

[12] Weiss G. (Ed.). *Multiagent Systems*. Cambridge, Mass., USA: MIT Press, 2000.

[13] Luck M. and d'Inverno M. *Understanding Agent Systems*. Berlin: Springer Verlag 2001.

[14] Trencansky I. and Cervenka R. Agent Modeling Language (AML): A Comprehensive Approach to Modeling MAS. Informatica, vol. 29, no. 4 (2005) pp. 391-400.

[15] Padgham L., Parker D., Müller J. and Parsons S. (eds.), Proc. of 7th Int. Conf. on Autonomous

Agents and Multiagent Systems. AAMAS 2008, Estoril, Portugal, 2008.

[16] Rao A. and Georgeff M. BDI Agents: From Theory to Practise. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95)*, pages 312-129, San Francisco, CA, USA, 1995

[17] Mild A. New Product Development Using Adaptive Neutral Agents in an Artificial Factory. A Virtual Experiment. Facultas 2001.

[18] Pechoucek M., Vokrinek J. and Becvar P. ExPlanTech: Multiagent Support for Manufacturing Decision Making. IEEE Intelligent Systems. 2005, vol. 20, p. 67-74.

[19] Pape U. Agentenbasierte Umsetzung eines SCM-Konzeptes zum Liefermanagement in Liefernetzwerken der Serienfertigung. Universität Paderborn: HNI, 2006.

[20] Martinez Lastra J., Lopez Torres E., and Colombo Armando W. A 3D Visualization and Simulation Framework for Intelligent Physical Agents. In: *Holonic and Multi-Agent Systems for Manufacturing*. Springer 2005.

[21] Jennings N.R., Bussmann S. Agent-Based Control Systems. In *IEEE Control Systems Magazine, June 2003*, S. 61-73.

[22] Huang G., Huang J. and Mak K. Agent-based workflow management in collaborative product development on the Internet. *Computer-Aided Design 32 (2000)*, p.133–144.

[23] Tseng K., El-ganzoury W. and Abdalla H. Integration of Non-networked Software Agents for Collaborative Product Development. *Computer-Aided Design and Applications Vol. 2 (2005),* Nos. 1-4.

[24] Mild A. and Taudes A. An agent-based investigation into the new product development capability. *Computational & Mathematical Organization Theory, Volume 13, Number 3, (2007),* pp. 315-331.

[25] Zhang X., Luo L. and Duan S. Evaluation of Product Development Process Using Agent Based Simulation. In: *Lecture Notes in Computer Science, Computer Supported Cooperative Work in Design IV: 11th International Conference, CSCWD 2007*, Melbourne, Australia, April 26-28, 2007.

[26] Chami M., Seemüller H. and Voos H. A SysML-based integration framework for the engineering of mechatronic systems. In *IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications MESA 2010,* p. 245–250.

[27] Stetter R., Voos H. AGENTES - Agent Based Engineering of Mechatronic Products. *In Proceedings of the 8th International Symposium on Tools and Methods of Competitive Engineering (TMCE) 2010.*

[28] Holt J. and Perry, S. *SysML for Systems Engineering.* Professional Applications of Computing Series 7, Institution of Engineering and Technology Press, UK, 2008.

[29] Object Management Group OMG, XMI Specification V.2.1.1, 2007.

[30] Object Management Group OMG, SPEM Specification V.2.0, 2008.