Potential of Dynamically Adaptable Simulation Models for Virtual Commissioning

Philipp Puntel Schmidt*, Alexander Fay

Helmut-Schmidt-University, Insitute of Automation, Holstenhofweg 85, D-22043 Hamburg
*philipp.puntelschmidt@hsu-hh.de

Simulation Notes Europe SNE 25(2), 2015, 59 - 68
DOI: 10.11128/sne.25.tn. 10291
Received: August 15, 2015 (Selected ASIM STS 2015 Postconf. Publ.); Accepted: August 20, 2015;

Abstract. Virtual commissioning (VC) is used to test control code deployed on Programmable Logical Controllers. Simulation models of a plant are the core of any VC approach. Simulation models should represent the plant in a way so that the correct process execution can be tested under customers' conditions. Simulation models of a plant are usually not built monolithically, but by many partial simulation models that represent the modules or components of the investigated plant. To ensure that the VC is efficient and provides helpful results, these partial simulation models can be implemented at different levels of detail, depending on the current test scenario. Usually, the definition of the modules' and components' level of detail is fixed. However, situations exist where more than one level of detail can be adequate. A dynamically adaptable level of detail seems beneficial to e. g. keep computing time at a reasonable level and to ensure meaningful results of the plants simulation model. However, no method or approach exists so far to handle a dynamically adaptable level of detail. The paper presents the research results of the authors on virtual commissioning and focuses on a simulation point of view and is organized as follows: In Section 1, a brief description is given on how to define the right granularity of simulation models used for virtual commissioning. Based on these results, several levels of detail and model types that can be used for a VC approach are introduced in Section 2. In Section 3, situations are described where more than one level of detail is suitable. In Section 4 and Section 5, potentials and challenges of a dynamically adaptable level of detail are dicussed and possible solution contributions that could yield benefits for a VC approach are shown.

Introduction

Simulation models that represent the investigated plant are the basis of any virtual commissioning (VC) approach. Usually, these simulation models represent specific elements of a plant like modules and components [1] and are built by many partial simulation models. From a mechatronic point of view [2], these partial simulation models simulate modules that can be seen as noteworthy elements of the overall system and be described as substantial function holders.

Modules realise specific processes or tasks within the plant and bundle their capabilities together to perform the plants greater purpose. Modules themselves are comprised of (hierarchically lower and granular finer) components like sensors, actors and others (Figure 1). Components as typical elements of a module perform - in cooperation and together with an appropriate controller - the specific process/task of the module.

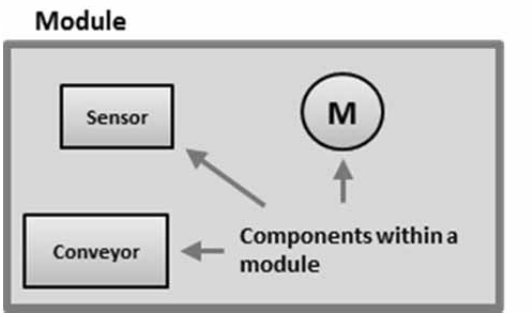

Figure 1: Components within a module.

The granularity of a simulation (what components or modules of a plant are modelled at which level of detail by which model types) takes a high impact on the simulation itself. The more variations of specific elements exist (which widens solution space n), the more difficult it is to realise a simulation model. This applies particularly to an adaptive level of detail, where by principle n elements can be described by m models.

While the number of different model types at different levels of detail and thus the overall number of models increases, kind and quantity of interfaces between these models increase as well (see also Section 2.1). As stated by Haberfellner et al. [3], the overall simulation model becomes consequently more detailed by increasing the number of interfaces. Rabe et al. [4] refers to Chwif et al. [5] and Robinson [6], stating that the intricacy of a simulation model is highly depending on the used level of detail. Eventually, a high and complex level of detail combined with a large amount of possible model types increase the necessary effort on validation of partial parts as well as of the entire model.

As a conclusion, only few specific parts of a manufacturing plant should be simulated at a high and complex level of detail, depending on the specific use case. The largest part of a manufacturing equipment model should be kept as easy as possible, easy to validate. Concerning the level of detail, the following rule should apply: As abstract and less complex as possible, as complex and detailed as needed [7]. Number and kind of specific levels of detail and model types should be restricted.

1 Defining the Right Granularity of Simulation Models

To approach a systematic definition of sensible lev-els of detail for simulation models used for VC, the appropriate granularity of simulation models should be clarified by means of appropriate model types. Auto-mated plants are structured hierarchically, describing the structure and arrangement of typical elements like sensors, actors and others. Accordingly, a hierarchical perception on plants can be promising. From operator to operator plant hierarchies might be distinguished in detail, which is why a universal, holistic view on plants is not possible without further investigation. Essentially, the well-known systems engineering approach on hier-archy within a system can be determined as a standard [3] when investigating structure and arrangement of automated plants and its involving elements. Furthermore, the data model of the digital plant regarding to VDI4499 [8] should be given one's careful consideration as appropriate database for elements or objects to be identified when looking at a plant's hierarchy and, in the end, architecture.

Therefore, as a prerequisite, the systems' architecture should be processed in a way so that further investigation is possible. According to Systems Engineering standards ([3], [9]) it is common to decompose a system into smaller elements like subsystems, modules and components. Components describe the smallest elements of a plant worth to be considered [28] and are not being split into granularly finer parts. Components represent typical resources like sensors, actors, conveying belts and others. They do not perform processes on their own but in cooperation. Strong interacting components can therefore be described as typical function holders within the automated plants. These function holders in form of (granular more roughly decomposed) modules perform the customers' defined processes based on the interaction of the planned and dimensioned components. Modules can therefore be seen as to be identified parts of the plant that should be individually simulated at an appropriate level of detail. Thus, in preparation to further investigation, it is important to identify modules of a system that can be mapped onto the implemented processes. The first step should be to investigate the systems' architecture.

To identify modules as substantial function holders within the plant, the system architecture DSM, comprising of components and their interrelationships, can be used [10]. The DSM is represented as an n-square (N^2) matrix where components label the rows and columns and are '[…] set in relationship to each other by entering marks or values in the DSM cells' [9]. The relationship can either be binary or, preferably, numerical [10]. Numerical values are advantageous as they allow to weigh the interactions. In this context, Pimmler and Eppinger state that '[…] the number and definitions of the interaction types is dependent upon the context of the given design problem' [10]. The basic procedure for building a system architecture DSM model is described in detail by Eppinger and Browning in [9] and is not further discussed here. Based on the identified interaction of components, several analysis methods on the created DSM model can be used. According to Eppinger and Browning, '[…] the most common method of analysis applied to [system] architecture DSM models is […] clustering' [9]. Clustering allows for the identification of interacting components in form of clusters that are described as 'a set of components grouped because of certain relationships, suggested through analysis of the [system] architecture DSM, and defined to comprise a module […]'.

Figure 2 shows this principle: Starting with a set of interacting components, modules can be derived by using the clustering algorithm.

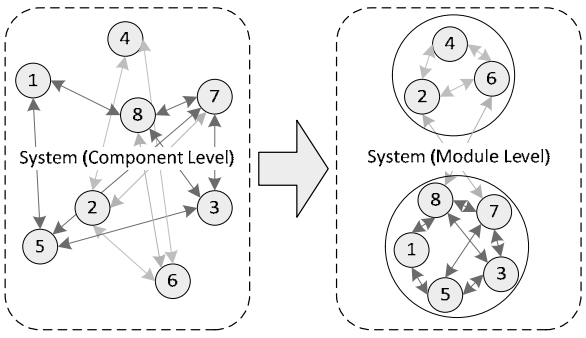

Figure 2: Principle of 'clustering': Components with strong interaction are arranged to clusters (which henceforth form the modules).

By applying the clustering algorithm, modules as substantial function holders can be identified from the system's architecture. Each module usually has numerous and different internal interfaces between the components which it comprises, but only a comparable small number of external interfaces. The latter allow the connection of several modules that altogether represent the investigated plant. The higher the number of interfaces isthe more complex a system is [3]. To keep the complexity of a module at a reasonable level, all components within a single module should therefore have the same level of detail in simulation. The different modules, however, will be modelled in appropriate, i.e. possibly different, levels of detail. This will reduce the modelling effort, especially in case of an automated model generation.

2 Model Types and Levels of Detail for Virtual Commissioning

Simulation models of plants' modules and components must be sufficiently meaningful with regard to the respective test case [3]. Models can naturally only be abstractions and simplifications of reality and can therefore only show selected aspects of a system's behaviour. A systematic classification on what aspects should be simulated is beneficial for a VC approach. As defined in [1], four levels of detail (*LoD*) can be identified for a (discrete manufacturing) VC approach and can be described as follows:

- Macroscopic LoD: Rough granular perception of the respective plant; modules are represented by a single simulation model (without comprised components). Only time based behavior is considered, goods that are transported and/or processed are not considered.
- Mesoscopic (not dynamical) LoD: Fine granular perception of the plant; the module's function is simulated by the interaction of simulation models of components that comprise the module. These components can be devices (like sensors or actors or frequency inverters) or mechanical/pneumatical/hydraulical systems (also called basic system according to [2]). Only time based behavior is considered, goods that are transported and/or processed are considered but movement is restricted to predefined paths (no free movement in space or collisions possible).
- Mesoscopic (dynamical) LoD: Fine granular perception of the plant as in the mesoscopic (not dynamical) LoD. Components show physical/dynamical behavior, where applicable. Goods that are transported and/or processed are considered but movement is restricted to predefined paths (collisions on these paths are possible, in contrast to mesoscopic (not dynamical) LoD).
- Microscopic LoD: Fine granular perception of the plant as in the mesoscopic (dynamical) LoD. Components as well as transported goods show physical/dynamical behavior. Movement of goods is possible in free space, including any type of collisions like interfering contours, etc.

Several model types (exemplary defined in [1], see also [11]) can be distributed over these four levels of detail and define the solution space used to build the respective plant's simulation model. These model types differ in aspects like time-based and dynamical/physical behavior. The specified model types can be described as follows:

- Dead time models (macroscopic LoD): The first model type to be defined can be the simulation of the mechanic and electronic domain in only one model, covering the hierarchical module level. The behaviour of the module is described here as a whole, individual components and therefore concrete domains of a mechatronic system are disregarded. From a system point of view this corresponds to a coarse-granular modelling approach. Models can be seen as black-boxes, they describe a mechatronic system (module)

ignoring its inner structure. By definition the function and behaviour is covered by only one model, this can be done by a simple transformation of the control input to control signals delayed only by a dead time (dead time models). Dead time models on module level can be modelled as simple data flow models since no physical characteristics are considered. Parameterization should be kept flexible, which means that dead time itself could be a parameter or is being calculated e. g. from the transport speed and length of a conveyor. Dead time models are only partly appropriate to describe complex correlations, since models would become very complex and unclear.

Goods or items that are transported or machined are not considered with this model type. Modelling dead time models on module level depend on system knowledge to picture the respective function and behaviour [12]. System knowledge describes the knowledge needed to describe circumstances and processes as a whole. Faults concerning runtime monitoring (e. g. a transport or a process does not finish in time) or discrepancy errors (e. g. errors that describe discrepancies on sensor signals) are to be modelled in an appropriate way and must be suitable to fit deterministic and stochastic investigations.

- Simple device and kinematic models (mesoscopic not dynamical LoD): Consideration of models on module level is often not sufficient, particularly when behaviour and function cannot be modelled in an appropriate way or only with high modelling efforts. A view on the mechanic and electronic domains of a mechatronic system might be necessary. This could be done by considering the component level as identified, where behaviour and function and therefore the individual characteristics of a mechatronic system/module are defined by collaboration of specific components. From a system point of view this corresponds to a fine-granular modelling approach, modelling the behaviour and functions of concrete components. In this particular case, models can be seen as white-boxes, describing the inner structure of a mechatronic system. For the electronic domain, this could be done similarly to module levels models as simple dead time models (named simple device-models). Depending on parameterization, simple device-models show behaviour and function of a device but offer no extra functions or physical behaviour. Simple kinematic models, describing movement and behaviour without moving forces, are the counterpart of simple device models and represent the model of the basic system. Movement is based on stated paths, whereby movement can be a translation, rotation or combinations of those (defined by the degrees of freedom of the basic system). Simple device- and kinematic models can, as well as dead time models, be modelled as simple data flow models since no physical characteristics are considered. Faults are to be modelled in an appropriate way.
- Complex device and kinematic models (mesoscopic dynamical LoD): If concrete physical behaviour is needed to describe the characteristics of the mechatronic system/module, models of components must provide functions containing equilibrium of power and moments. For the electronic domain, this can be done by models that cover flow and potential of physical systems (as device-models). Simple physical models considering forces that cause movement are the matching models of the basic system. As the previously defined simple kinematic models, movements are limited to stated paths, covering both translational or rotational movements and combinations of those, on this occasion described physically/dynamically. A free movement within space is excluded here. These models are the corresponding partners for device models (named kinetic models), representing the mechanic domain. Faults are to be modelled in an appropriate way. The modelling itself is preferably done by an object-oriented non-causal modelling language suitable for dynamic simulation
- Kinetic models (microscopic LoD): When free movement in space is necessary to cover the characteristic of the mechatronic system/module and its covered process, movements on stated paths are not sufficient anymore. Models must be able to emulate reality in a way that use cases like possible collisions between objects and other structural elements are simulated in a proper way. Models of the basic system therefore depend necessarily on geometric design data [13]. Movement of goods and items is subject to physical rules (complex kinetic models). To simulate physical behaviour of the basic system, models of the device must be simulated in an equivalent level of detail. Device-models as already defined are not suitable, since undefined behaviour of the complex kinetic models could be the consequence when building a simulation model with these models. Device models must be able to react properly to complex kinetic model behaviour, indicating that

complex device-models are needed for the electronic domain. Complex device-models must provide behaviour like switching hysteresis of sensors or s-curves of variable-frequency drives. Functions like the s-curves should be emulated in an appropriate way. As already stated for physical based models, modelling itself is preferably done by an object-oriented, non-causal modelling language, suitable for dynamic simulation. Faults are to be modelled in an appropriate way.

Table 1 shows an overview of the defined levels of detail as well as several important aspects of the particular levels of detail that are important for a VC approach.

Aspect:	Macroscopic	Mesoscopic dev. (nd)	Mesoscopic dev. (d)	Mesoscopic mech. (nd)	Mesoscopic mech. (d)	Microscopic dev.	Microscopic mech.
Consideration of entire process mapping	✔	✔	✔	-	-	✔	-
Time-based behaviour	✔	✔	X	✔	X	X	X
Dynamical/physical behaviour (if applicable)	X	X	✔	X	✔	✔	✔
Failure states	✔	✔	✔	✔	✔	✔	✔
Consideration of processed good:*	X	-	-	✔	✔	-	✔
- Movement of goods on stated paths (= degrees of freedom)	X	-	-	✔ **	✔	-	X
- Free movement of goods in space	X	-	-	X	X	-	✔
- Collision of goods possible	X	-	-	X	✔ ***	-	✔
dev: device, mech: mechanical /pneumatical/hydraulical basic system (see VDI2206 [2]), nd: not dynamical, d: dynamical							

** Good is synonym to bulk goods or piece goods*
*** Acceleration free movement*
**** Only possible in simulated degree of freedom*

Table 1: Overview of defined levels of detail and several important aspects of VC.

2.1 Ensuring consistent interfaces between simulation models in different levels of detail

Different levels of detail indicate different modelling techniques. Time based models on a macroscopic resp. mesoscopic (not dynamical) level of detail can be modelled using a causal modelling approach, while simulation models that show dynamical behaviour (mesoscopic dynamical, microscopic level of detail) usually follow a a-causal modelling approach, as already mentioned in the previous section.

However, simulation models in different modelling approaches are not necessarily compatible regarding to their interfaces: a consistent data flow is not possible at all times and must be ensured, especially in case of an automated model generation ([24], [25]) where no manual intervention or correction is demanded.The definition of modelling regulations, e. g. presented by a set of rules, allows for a systematic identification of situations, where additional simulation models like coupling or termination elements (that must be included in the simulation library, see [14] and [27] for an example) are necessary to ensure the interoperability of the partial simulation models. A detailed description on how to solve the problem of inconsistent interfaces is given in [14] and is not further discussed here.

2.2 Deficits when defining the level of detail for certain modules

As described before, each single module can be seen as a substantial function holder within a plant [1]. These function holders perform the customers' defined processes.

Based on available engineering data and heuristics that consider the practical knowledge of the engineers involved, a situation related (test-case specific) required level of detail can be assigned to each single module (see Section 1, also [15]). At the moment, this assignment isfix for the simulation run and cannot change. However, modules often not only perform one process, but many, depending on the character of the appropriate module. Situations can arise where modules can be modelled in multiple levels of detail, according to the appropriate situation and the process that is being executed. A module e. g. executes three different processes where each process requires a different level of detail. An example will be given in the next section.

3 Identifiying Situations where an Adaptable Level of Detail is Appropriate

As a first step towards an adaptable level of detail, situations should be systematically identified where the approach is beneficial. An easy transport processes shall be given as example to demonstrate situations where an adaptable level of detail is appropriate.

3.1 Example

A work piece is to be transported over 2 conveyors (CV01, CV03) and a turntable with integrated conveying belt (TT02) there and back again (Process steps P1 to P6 in Figure 3).

All conveying belts consist of one motor; the turntable contains an extra motor for rotation movement. Conveying belt CV01 and CV03 have one sensor each (S1 placed at the beginning of CV01 and S6 placed at the end of CV03) for (accurate) position recognition. Turntable TT02 has 2 sensors (S2 and S3) on its conveying belt. The position of the (rotating) turntable is recognized by two mechanical switches (S4, S5).

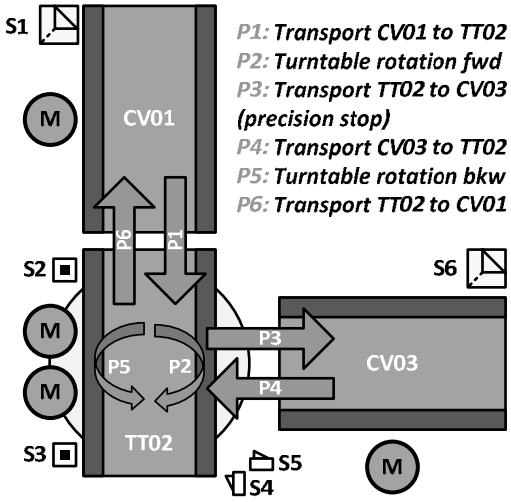

Figure 3: Example as described.

While sensors S2 and S3 can be described as standard capacitive sensors, sensors S1 and S6 allow for precise positioning (yellow shaded, laser distance sensors). Performed processes are P1 to P6 (Figure 3, green arrows).

P1, P2, P4 and P5 can be described as 'standard' conveying processes or 'technical cycles' that do not have any special requirements e. g. regarding to positioning accuracy. Process P3 and P6 however need very accurate positioning (± 0,1mm) for further processing of the work piece (not part of this example model). Figure 4 shows an excerpt of the PLC program that exemplary shows execution of process steps P1 and P2.Eventually, a situation related required level of detail can be defined based on the processes. When performing process P1, P2, P4 and P5, no further requirements on the simulation emerge – the usage of a macroscopic level of detail (dead time models) seems to be adequate to test the programmed PLC-functions (see also Figure 4).

Following this idea, two levels of detail, depending on the process step, can be suitable for conveyors CV01 and CV03: When P1 and P4 are executed, a macroscopic level of detail (and the included dead-time-model) is sufficient.

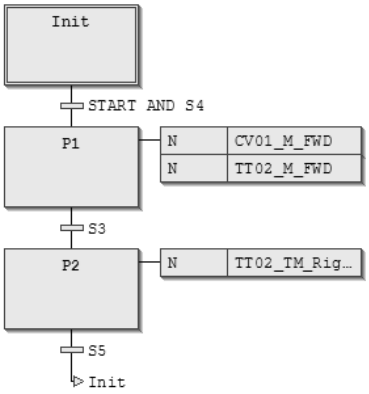

Figure 4: Exemplary PLC program that shows execution of process steps P1 and P2.

However, process P3 and P6 indicate that accurate positioning must be ensured. Parameters like ramps and physical effects like acceleration and friction become important elements of the module models in these process steps, as they influence the quality of the simulation results and, finally, the quality of the PLC-Code itself. The needed physical behaviour can be simulated by a mesoscopic (dynamical) level of detail (complex device and kinematic models, see Chapter 1).

3.2 A systematic way to identify modules that can potentially switch regarding to their level of detail

In case several levels of details are possible, from all suitable levels of detail often the highest and most complex one is chosen to ensure correct behaviour of a module in any case. This can lead to situations where a high and complex level of detail is useless for most of the simulation run since a low (curtailed in the meaning of functionality) level of detail would be sufficient for most of the time. Situations are possible where 1) the computing time increases to a level that prevents VC from running smoothly and 2) where effort on validation can increase to a level where the overall effort/benefit ratio of a simulation is compromised.

An adaptable level of detail where simulation models can switch through different levels of detail (depending on the test case) can be promising. However, there is no known solution for this challenge at the moment. To describe the issue of switching simulation models (regarding to their level of detail) in a standardized and systematic way, the Domain-Mapping-Matrix (DMM) [16] can be used as a supporting tool. The DMM, represented by a rectangular (n x m) matrix, can combine two different domains (in this consideration resources and processes). Rows are represented by appropriate modules (representing the resources of the system investigated) while columns are filled with the process steps that are executed by the modules.

For each module which is involved in a particular process step, the cell at this intersection must be filled with a score. The mapping process is executed by entering a value that represents the strength of the interaction of designed automated processes and processing resources selected during the engineering process itself. In context of the given design problem, each conceivable interaction must be scored.

Depending on the simulation systems' supported levels of detail, the overall quantification scheme of the score could be based on the amount of levels of detail available for modelling. If e. g. four levels of detail (see Section 1) are provided, the scores' values could be based on a range from 1 to 4, in reference to each available level of detail. Therefore, the numbers present the overall needed level of detail (where 1 is defined as macroscopic and 4 as microscopic), ascertained by engineers. Figure 5 shows an example on how the interactions of modules and processes (based on the example in Chapter 2.1) could be weighted.

DMM

	Process 1	Process 2	Process 3	Process 4	Process 5	Process 6
CV01	1					3
TT02	1	2	1	1	2	1
CV03		!	3	1		

Figure 5: Example of a Domain Mapping Matrix where processes are assigned to modules within a plant. Processes refer to process steps as defined in Chapter 2.1.

Attention should be paid to situations where a module has different scores (exemplarily shown by the red boxes in Figure 5). This can be identified by examining the modules associated row. As a conclusion, the DMM indicates in an easy way to identify situations where more than one level of detail is appropriate. As already described, the highest (and most complex) level of detail of all possible is often chosen for the simulation run to ensure correct behaviour of a module in any case (max() function over the module's associated row), despite useless for e. g. CV03 when performing process P4 or TT02 performing Processes P1, P3, P4 and P6.

4 Potentials and Challenges of an Adaptable Level of Details

A second step in introducing an adaptable level of detail should focus on possible implementation strategies regarding to the used simulation tool and overall simulation framework and procedure. The main benefit (lower computing time) of an adaptable level of detail should be in focus.

4.1 Possible implementation strategies

Simulation is always strongly depending on the used simulation tool as well as framework and the overall simulation approach like discrete event simulation or continuous simulation. The computing time of a simulation is influenced by multiple factors according to [17]: The type of the used numerical solver (for a detailed comparison of common solvers see [18]), amount of continuous state variables as well as the sheer quantity of events that must be handled by the solver [19]. If an adaptable level of detail is demanded, two very basic implementation strategies can be identified.

1. Implementation in one single tool (e. g. in Modelica [20]): One tool provides all levels of detail in e. g. a (copious) library. The tool must be able to handle discrete event simulation as well as continuous simulation approaches. The tool must be able to swap (partial) simulation models during runtime. This is also called 'integrated simulation' [17].
2. Implementation in different tools according to tool specific abilities and strengths (e. g. Modelica [20] and Matlab/Simulink [21]): Several libraries in different modelling tools provide objects in a specific level of detail. Matlab/Simulink could e. g. be used

for dead-time models (since dead time models have a discrete event character) while a Modelica library can provide models where physical/dynamical behavior is needed.

A middleware (like the Functional-Mockup-Interface (FMI) [22] or implementations with respect to the High-Level Architecture (HLA) [23] standard) is necessary to interconnect these (partial) simulations. The middleware must be able to swap (partial) simulation models during runtime. This principle is also called 'separated simulation' according to [17].

Both implementation strategies have their advantages and drawbacks: When implementing all levels of detail feasible in just one single simulation tool, it must be capable to handle both discrete (macroscopic, mesoscopic not dynamical) and continuous (mesoscopic dynamical, microscopic) behaviour ('Hybrid-Modelling', compare [17]). Additionally, the simulation tool must provide (numerical) solvers that are capable of handling not only discrete event simulation or continuous simulation, but both. Cardinality of the included simulation library is estimated to be very high, exacerbating servicing as well as model building effort (regardless if models are built by an automatic model approach [24], [25], or manually).

Segregation of discrete and continuous behaviour in different tools (e. g. Matlab/Simulink for discrete event simulation, Modelica for continuous simulation) requires a middleware with standardized interfaces to ensure interoperability of the respective models. Liu and Frey [17] describe this as an intermediate communication and synchronization layer. The well-known Functional Mockup Interface (FMI) [22] is one approach that follows this idea and has already found widespread acceptance in industrial applications.

FMI was introduced with the intention to deliver an interface to develop complex systems where different parts of the system can be modelled in different simulation tools. Simulation models are provided in form of a Functional Mockup Unit (FMU) (exported by the respective simulation tool) and are implemented in the standardized framework. However, FMI increases the complexity of the simulation approach to a level that could make an automatic generation approach according to [24], [25] unfeasible. Efforts on parameterization as well as adjustment of the FMI framework itself are considered to be rather high.

4.2 Influences on computing time when simulating in different levels of detail

Different levels of detail (precisely: different model types) can be indicative of different workloads needed to calculate the plant simulation model, being one of the main reasons an adaptable level of detail might be suitable to avoid unnecessary high computing time. A highly detailed simulation model indicates, due to the high amount of events to be calculated, that computing time may increase to a level that prevents VC from running 'smoothly'. An example on how smoothness can be defined is provided by [26]: A fixed step of 1ms (hard real-time) of the simulation must be ensured so that the PROFINET communication from simulation model to the PLC is not corrupted within a VC test bench.

As already mentioned, the numbers of triggered events have a large impact on computing time of a simulation model. Each time an event is triggered (e. g. when a sensor triggers or control variables within the PLC change), the deployed solver restarts calculating the whole simulation model. Depending on the solver, these calculations can be iterative (called event-iterations) to ensure the specified simulation accuracy. This increases computing time even further. More complex, very detailed simulation models (that consider physical/dynamical behavior) have naturally a high amount of events and need potentially more computing time than an easy, time based simulation model which is based only on dead-time models (that should only have a low number of events). The defined models types (Chapter 1) should therefore not only differ in the ability to simulate time based and physical/dynamical behavior, but also mainly in the number of events they potentially generate. Furthermore, the used simulation models should be highly optimized to a specific solver (e. g. in [26], the Euler solver is used) to ensure a hard real-time at any time whenever needed.

4.3 Conclusions and thoughts about an adaptable level of detail

As a conclusion, the used simulation models, e. g. stored in a model library (see [27] for further details and an example), should be optimized in a way so they generate as less events as possible. Additionally, it must be ensured that the number of events grows with the respective level of detail and not vice versa: models at a macroscopic level of detail should generate (clearly) less events than models at a microscopic level of detail.

This highly correlates with the solver used and must be taken into consideration when building or optimizing a simulation library that contains models at different levels of details. The main benefit and the ultimate goal of an adaptable level of detail, is therefore mainly depending on the events generated and the solver used, not necessarily from the simulation principle itself (discrete event/continuous simulation). Additionally, the interoperability of the simulation models must be ensured at all time, not only when the level of detail is determined static, but especially when simulation models should change during runtime.

On the technical side, no possibility to change models during runtime exists, to the authors' best knowledge, so far, and this holds for both implementation possibilities (integrated/separated simulation). This can be seen as a major point that should be taken into consideration in future investigations.

5 Requirements for Implementing an Adaptable Level of Detail

As concluded in Chapter 3, the used simulation tool, simulation framework and number of events have a huge impact on the implementation of an adaptable level of detail. Several requirements can be identified that must be accomplished regardless of the used simulation environment. These requirements represent important factors that not only concentrate on the simulation model or the used framework, but also on the simulation (building and executing) process itself and shall be considered when an adaptable level of detail is to be implemented in future applications:

- It must be possible to generate the simulation model automatically: Information potentially necessary for a dynamical switching should not prevent an automatic generation approach (e. g. described by [24] or [25]).
- The level of detail may only be switched considering modules: Only complete modules (including all comprised components) may be switched through different levels of detail, not the components comprising a module (the goal is to keep complexity of the simulation model as well as intricacy of the switching process itself on a reasonable and manageable level).
- Interoperability must be ensured: Interfaces between partial simulation models should always be compatible, even when these models are switched between several levels of detail.
- Computing time shall not increase while 'switching' the simulation models: Effort on 'switching' simulation models to different levels of detail should be kept to a minimum regarding to computing time to ensure real time ability.
- 'Switching' simulation models within a process step must be prevented: A specific level of detail assigned to a module must be assigned fix until the process is terminated. Switching levels of detail is therefore not allowed within a process, but between two subsequent processes.
- It must be clearly identifiable what process is being executed: In case a module is able to execute several processes, the simulation tool/framework must be capable to identify the specific process.
- Models used for VC should always be optimized to generate the minimum possible number of events: The model library (an example can be found in [27]) must be optimized and it must be ensured that simulation models in a low level of detail create significant fewer events than models at a high level of detail. This can mean that models should be optimized to run with a specific solver in a specific simulation environment.

6 Conclusion

The paper presents considerations regarding an adaptable level of detail when running a simulation with the purpose of virtual commissioning. Furthermore, an outlook on how an adaptable level of detail might be implemented according to simulation library and simulation framework was given. While different levels of detail have already been implemented in several test cases [1], at the moment no known simulation tool as well as simulation framework is capable of switching simulation models during runtime of the simulation. Future work on the topic should be focused on two aspects: Firstly, a refined method to identify situations where an adaptable level of detail is suitable (based on the method introduced in this paper) and secondly a technical solution that enables swapping simulation models at different levels of detail.

References

[1] Puntel Schmidt P, Fay A. Levels of Detail and Appropriate Model Types for Virtual Commissioning in Manufacturing Engineering. In*: Proceedings MATHMOD 15 – Vienna Conference on Mathematical Modelling, Vienna, 17.-20. February 2015*.

[2] VDI2206: *Design methodology for mechatronic systems.* VDI-Verlag, Düsseldorf, 2004.

[3] Haberfellner R, de Weck O, Fricke E, Vössner S. *Systems Engineering. Grundlagen und Anwendungen.* Orell Füssli Verlag AG, Zürich, 2012.

[4] Rabe M, Spieckermann S, Wenzel S. Verification and Validation for Simulation in Production and Logistics. *Simulation News Europe.* 2010; 19(2):21-29.

[5] Chwif L, Pereira B MR, Paul RJ. On simulation model complexity. In: Joines JA, Barton RR, Kang K, Fishwick PA, *Proceedings of the 2000 Winter Simulation Conference*, Orlando (USA), IEEE, Piscataway, 2000. P. 449-455.

[6] Robinson S.*Simulation: The Practice of model development and use.* John Wiley & Sons, Chichester, 2004.

[7] Hrdliczka V. Leitfaden für Simulationsbenutzer in Produktion und Logistik. *ASIM Mitteilungen* 58, 1997.

[8] VDI4499: *Digital Factory*. VDI-Verlag, Düsseldorf, 2008.

[9] Eppinger SD, Browning TR. *Design structure matrix methods and applications.* Cambridge, Mass: MIT Press (Engineering systems), 2012.

[10] Pimmler TU, Eppinger SD. *Integration analysis of product decompositions.* Cambridge, Mass: Alfred P. Sloan School of Management, Massachusetts Institute of Technology (Working paper / Alfred P. Sloan School of Management, WP # 3690-94-MS), 1994.

[11] VDI3693: *Virtuelle Inbetriebnahme. VDI-Richtlinien-Entwurf.* VDI-Verlag, Düsseldorf, 2015.

[12] Lüdecke A, Pelz G. Top-Down Design of a Mechatronic System.In: *Third Forum on Design Languages (FDL 2000)*, P. 151-158. Tübingen, Germany, 2000.

[13] Reinhart G, Lacour FF.Physically based Virtual Commissioning of Material Flow Intensive Manufacturing Plants. In: *3rd International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV 2009)*. Munich, Utz, 2009.

[14] Puntel Schmidt P, Fay A, Konsistente Simulationsmodelle für die virtuelle Inbetriebnahme fertigungstechnischer Anlagen mit Hilfe regelbasierter Modellverbinder. In: Rabe M, Clausen U, editors. Fraunhofer IRB Verlag, Stuttgart 2015 – Tagungsband der 16. ASIM Fachtagung Simulation in Produktion und Logistik. *Simulation in ProductionandLogistics 2015,* Dortmund, 23.-25. September 2015.

[15] Puntel Schmidt P, Fay A. Applying the Domain-Mapping-Matrix to Identify the Appropriate Level of Detail of Simulation Models for Virtual Commissioning. In: *2nd IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control Schedule (CESCIT)*, Maribor, Slovenia, 2015.

[16] Pimmler TU, Eppinger SD. *Integration analysis of product decompositions.* Cambridge, Mass: Alfred P. Sloan School of Management, Massachusetts Institute of Technology (Working paper / Alfred P. Sloan School of Management, WP # 3690-94-MS). 1994.

[17] Liu L, Frey G. Efficient Simulation of Hybrid Control Systems in Modelica/Dymola. *Proceedings of the 6th Vienna International Conference on Mathematical Modelling (MATHMOD 2009),* Vienna, Austria, Feb. 2009. pp. 1344-1352.

[18] Liu L, Felgner F, Frey G. Comparison of 4 Numerical Solvers for Stiff and Hybrid Systems Simulation. *Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2010),* Bilbao, Spain, Sept. 2010.

[19] Liu L, Felgner F, Frey G. Modellierung und Simulation von Cyber-Physical Systems, *Proceedings of the 12th Fachtagung EntwurfkomplexerAutomatisierungssysteme (EKA 2012),* Magdeburg, Germany, May 2012. pp. 149-157.

[20] Modelica Association [Internet]. [cited 2015 May 05] Available from: www.modelica.org

[21] MATLAB/Simulink v8.4. Natick, Massachusetts: The MathWorks Inc., 2014.

[22] Modelisar, FMI for Model Exchange 1.0 Specification [Internet]. [cited 2010]. Available from: https://svn.modelica.org /fmi/branches/public/specifications/FMI_for_ModelExchange_v1.0.pdf

[23] IEEE 1516: *Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules.* 2010.

[24] Puntel Schmidt P, Fay A, Riediger W, Schulte T, Köslin F, Diehl S. Validierung von Steuerungscode mit Hilfe automatisch generierter Simulationsmodelle. In: *at – Automatisierungstechnik*. 2015; 63(2):111–120.

[25] Barth M, Fay A. Automated generation of simulation models for control code tests. In: *Control Engineering Practice.* 2013; 21(2):218-230.

[26] Riediger W, Puntel Schmidt P, Köslin F, Schulte T. Hardware-in-the-Loop-Simulation fertigungstechnischer Anlagen. In: *SPS/IPC/DRIVES 2014*, Nürnberg, Germany, 25. -27. Nov. 2014.

[27] Köslin F, Puntel Schmidt P, Riediger W, Fay A. Entwurf einer Modelica Simulationsbibliothek für die virtuelle Inbetriebnahme fertigungstechnischer Anlagen. In: *7th International Symposium on Automatic Control, AUTSYM 2014*, Wismar, Germany, 25.–26.09.2014.

[28] Göring M, Fay A, Automation Systems – Formal Modeling of Temporal Change of Physical Structure. In: Proceedings of the 'IEEE IECON 2012, the *38th Annual Conference of the IEEE Industrial Electronics Society'*, Montréal, Canada, 2012.