Simulation Based Implementation of Flexible Task Oriented Robot Controls Using the System Entity Structure and Model Base Approach

Tobias Schwatinski, Thorsten Pawletta, Sven Pawletta
{tobias.schwatinski, thorsten.pawletta, sven.pawletta} @ hs-wismar.de
Hochschule Wismar – University of Applied Sciences: Technology, Business and Design
Research Group Computational Engineering and Automation

Contents

This paper introduces a new method used for the development of Flexible Task oriented robot Controls (FTC) using the System Entity Structure (SES). Task oriented robot controls are based on the composition of atomic tasks with the aim of achieving a previously specified goal. Flexible task oriented controls differ in that the composition of atomic tasks is not predefined fixedly but is composed during the operation of the control on basis of actual process states and with respect to any constraints according to the sequence of tasks. The System Entity Structure is an ontology, which can be used for the hierarchical representation of existing or imagined systems. It is shown how to automatically generate and execute FTCs for cooperating robots specified by a SES and an associated model base (MB).

1 Introduction

Flexible Task oriented robot Controls (FTC) consist of several atomic tasks which are composed with respect to any constraints of their sequence with the objective to achieve a specific goal. The concrete sequence of atomic tasks can be determined either on the basis of actual process states during control operation or on the basis of predictive process simulations. Therefore FTCs belong following [7] to intelligent robot controls and are related according to their implementation due to requirements and complexity to the "large-scale" development in [1]. As a consequence the implementation of such robot controls have to be realized following a systematic development process. This paper presents the FTC/SES method used for the systematic development of Flexible Task oriented robot Controls (FTC) on the basis of the System Entity Structure (SES) and Model Base (MB) formalism. The SES is a basic element of the FTC/SES method. The SES was originally developed in the eighties by Rozenblit and Zeigler and has been continuously enhanced to data engineering [3] till this day. The SES is an ontology that is designed for the hierarchical representation of real or imagined systems and is mostly used for the definition of meta models within the field of simulation technique. In our research the SES is used for the specification of flexible industrial robot controls including the subordinated controlled process. By means of the SES the overall control task is divided in subtasks which are composed of atomic tasks and other composed

This work is supported by the German Federal Ministry of Education and Research (support code 1747X08).

subtasks. The SES-based modular and declarative specification of a control and the controlled processes supports besides a systematic development the re-usability, adaption and maintenance of controls. Moreover, the FTC/SES method is based on the Simulation-Based Control (SBC) approach [2] and supports the successive development of simulation models within a homogeneous computing environment beginning from the design phase till the operation phase. In the following the basics of the SES and the SBC are shortly introduced. After that their combined usage for the specification of cooperating robot controls is discussed on the basis of an example. Subsequent the automatic generation of executable controls is shown. Finally, important aspects of a prototype implementation and some experiences concerning the introduced application are summarized.

2 Basics of the System Entity Structure and the Simulation-Based Control approach

2.1 The System Entity Structure

The System Entity Structure (SES) is an ontology. The SES forms a tree, whose nodes can be distinguished following [3] into four node types: (i) entity, (ii) aspect, (iii) multiple-aspect and (iv) specialization. The general sequence of nodes in a SES is pictured in figure 1 (a).

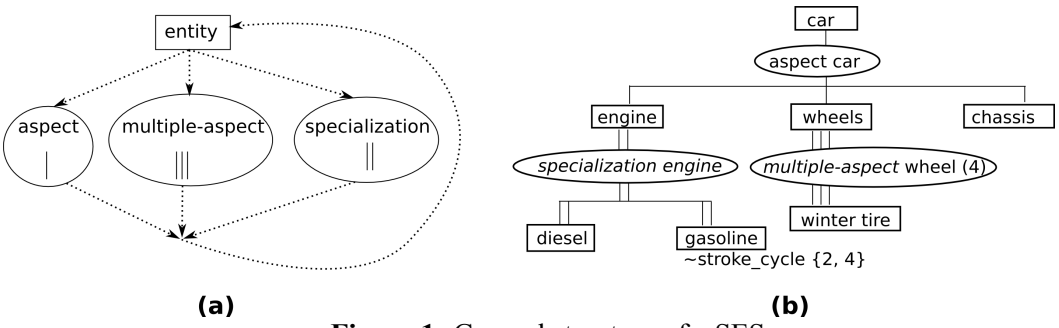

Figure 1: General structure of a SES

Entity nodes represent elements of the real or imagined world. Aspect nodes are used for the decomposition of entity nodes into more finer-grained structures. Multiple-aspect nodes define multiplicity of entity nodes and specialization nodes represent categories or families of characteristics of entities. In addition attributes and their domain of definition can be attached to any node. Figure 1 (b) shows exemplarily a SES for the specification of several automotive types. Every entity car consists according to the aspect node car of the entities engine, wheels and chassis. The entity engine is either specialized to the entity diesel or the entity gasoline. Moreover, the attribute *stroke_cycle* has been attached to the entity gasoline, which can be set with the values two or four. The entity wheels is followed by a multiple-aspect node with the parameter *multiplicity* = 4. Hence, this node will be break down into four winter tire entities. Therefore the SES in figure 1 (b) characterizes the three different automotive types:

- type 1: diesel engine, four winter wheels, chassis
- type 2: two stroke cycle gasoline engine, four winter wheels, chassis
- type 3: four stroke cycle gasoline engine, four winter wheels, chassis

In doing so the SES can be used for the clear definition of different characteristics of any composite system. Originally the SES has been developed for the specification of models in the field of simulation technique. The SES in combination with a model base (MB) that contains an executable software component for each leaf node of the SES allows a general automatic software generation. Besides all coupling relations between the entities have to be specified at aspect nodes. Moreover, the SES supports the specification of necessary sequences using constraints if several alternatives can be chosen. The simple example in figure 1 does not define any constraints.

2.2 The Simulation-Based Control Approach

The Simulation-Based Control approach (SBC) described in [2] is a specific type of the Software in the Loop (SiL) principle [4] and supports the throughout usage of simulation models during the whole development process of controls. Simulation models are stepwise enhanced beginning from design phase till the automation phase and finally are used as control software directly during the operation phase using an implicit code generation. This approach allows the usage of a development PC or an industrial PC for the control of real processes. The entire development process of controls based on the SBC is schematically shown in figure 2. The SBC defines that any control software consists of a control model, an interface and if required a process model. The interface provides the connection between the real process and the control software. This specific type of communication with a real process is called implicit code generation. The process model maps the behavior and states of real process components. In addition the process model that has been already used in the automation phase can be integrated into the control software as state observer. This procedure can increase the quality of controls, e.g. by calculating additional or immeasurable state values. The control model maps the complete control logic.

3 Development of robot controls based on a declarative specification

3.1 Integration of the SES/MB formalism and the SBC approach

The SBC approach supports an effective implementation of robot controls from the beginning of the design phase till the end of the operation phase using simulation models. The SES/MB formalism supports a systematic and declarative specification of dynamic systems by a tree structure (SES) and additionally an automatic program generation using predefined parameterizable modules from a model base (MB). In the following both approaches are used for the definition of task oriented robot controls and it will be shown how highly flexible controls can be implemented.

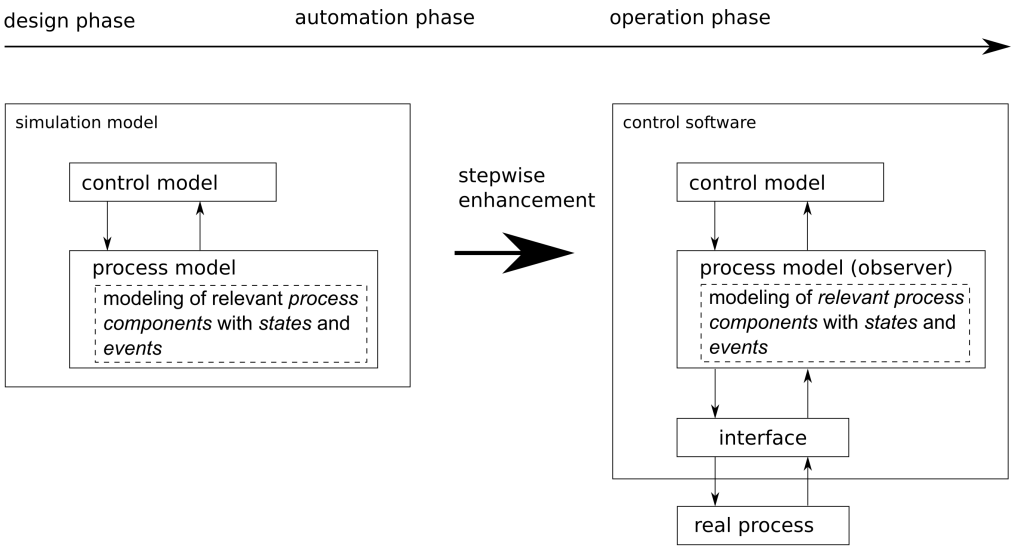

Figure 2: Simulation-Based Control approach (SBC)

The SBC supports following [2] the definition of task oriented controls. Predefined atomic tasks are composed and parametrized within a control model according to a specific control objective. In doing so any control commands and any reactions on states of the real system or the process model are programmed in atomic tasks. This basic principle is shown schematically on the basis of a simple control model in figure 3.

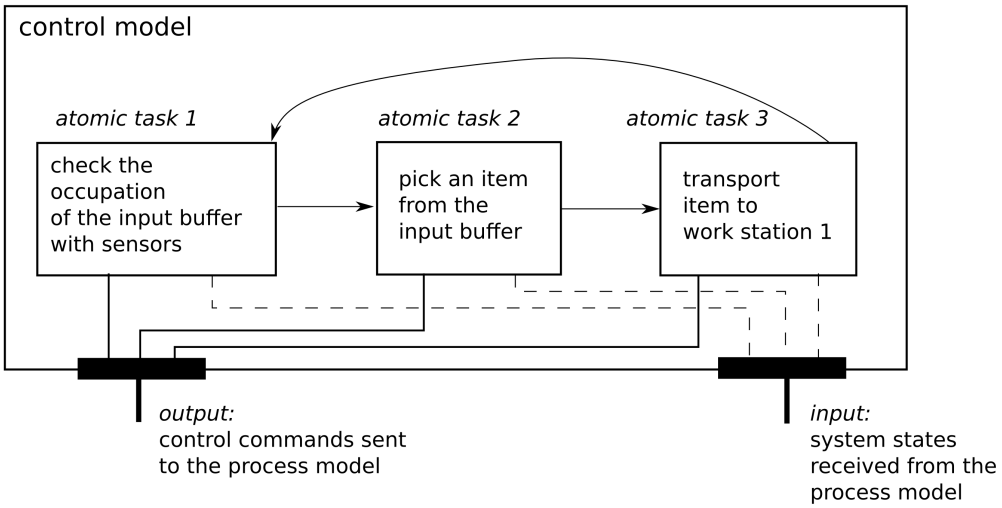

Figure 3: Coupling of atomic tasks in a control model following the SBC approach

Figure 3 shows that the sequence of atomic tasks is fixed within a control model. The whole flexibility of a control has to be implemented inside the atomic tasks and by means of their coupling relations. In particular complex and flexible robot applications comprise multifaceted relations between the atomic tasks within a control model as well as between the control model and the process model as a whole. This leads to high complex controls, because the SBC approach supports a structural oriented modeling in a software system but unfortunately it supports only a fixed composition of tasks.

The declarative specification of controls and the process-oriented generation of executable controls using the SES/MB formalism shall countervail this drawback of the SBC. The integration of the SES/MB formalism and the SBC approach for implementing Flexible Task based robot Controls is called FTC/SES method.

The most important elements and interactions of a flexible task oriented robot control following the FTC/SES method are pictured in figure 4. The SES specifies the structure and parameters of the whole system, e.g. the flexible control and the subordinated processes and interfaces. In our research the syntax and semantic of the original SES defined in [3] have been slightly changed and hence it is called control-SES. The control-SES specifies the all-embracing set of possible control variants. The synthesis and generation of a concrete, mostly temporary, control variant is performed by a task scheduler/control generator (TS/CG). The input to the TS/CG is a parameter vector that provides process based information for the SES analysis. The execution of the current control variant is performed by a control-PC. The control-PC is responsible for the communication with any actors, sensors and execution controllers of the real process and other control-PCs. Moreover, it updates system states and events of the current control variant. The task scheduler monitors every change in system states and the occurrence of any events during the execution of the current control variant. If predefined events, e.g. sensor signals, take place the current control variant will be interrupted. This leads to a modified parameter vector and consequently to the synthesis and generation of a new (temporary) control variant. This procedure is repeated till any predefined abort criteria take place. In particular the declarative specification of robot controls and the operation of the task scheduler are discussed in the following subsections.

3.2 Declarative specification of robot controls

In this research we propose a slightly modified SES formalism called control-SES for the specification of flexible industrial robot controls. The fundamental ideas are discussed by means of a small application. The application consists of two cooperating robots, each one with a separate buffer. The objective target for both robots is to arrange the objects in both buffers in conformity with a user defined order. To fulfill this objective the robots have to cooperate, because objects are stacked in the buffers. The total amount of places at the storage areas is much smaller than the total amount of objects. Every robot has to temporary cache objects from the other robot in its own input buffer so that the other one can operate its necessary sort sequence. The transfer of objects takes place directly between both robots.

At the beginning the control has to determine an optimal sort sequence to minimize the total amount of steps. A simplified control-SES of the described robot control is

shown in figure 5. For the reason of simplification the pictured SES includes only a small subset of the atomic tasks necessary to fulfill the described application.

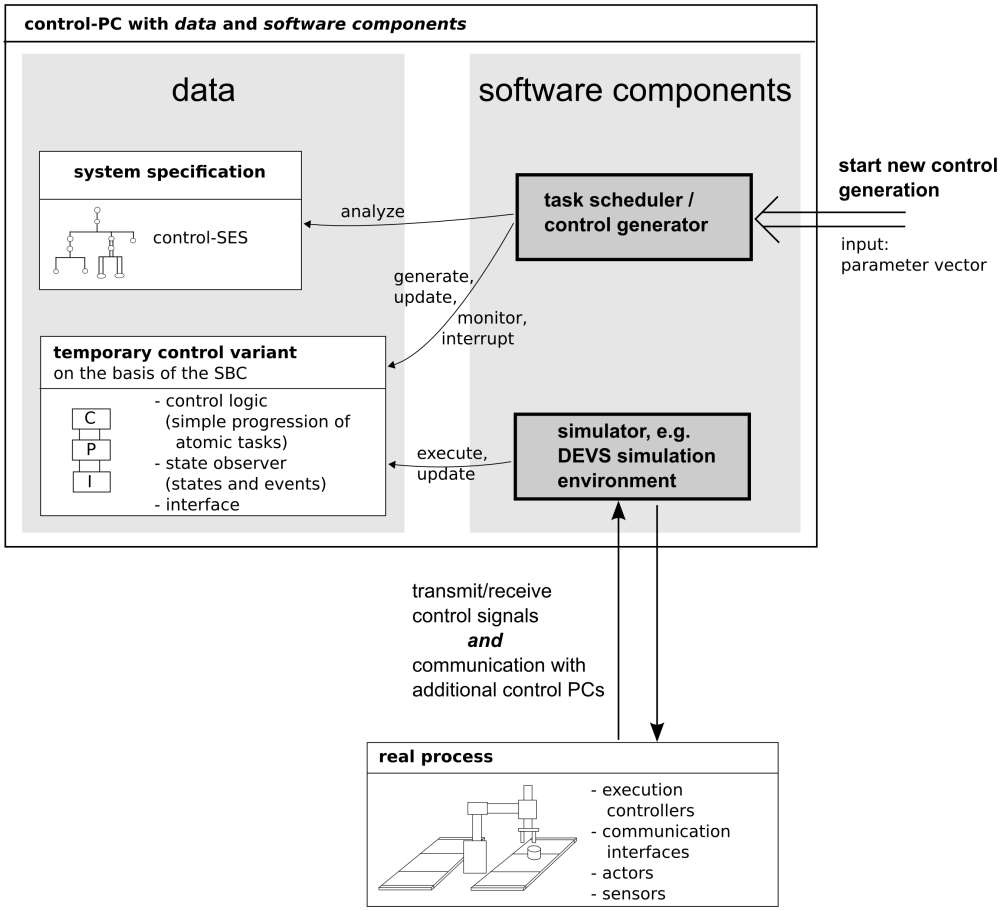

Figure 4: Elements and interactions of a flexible task oriented robot control following the FTC/SES method

The pictured control-SES consists of two parts. The upper part specifies the time invariant part of the control that defines according to the SBC a control model, a process model and a process interface. The overall task of the robot control is structured in several smaller tasks which are specified in the lower part. This part of the control-SES presents the time variant characteristics in terms of specialization nodes, which specify the alternative usage of atomic tasks to synthesize a concrete control variant.

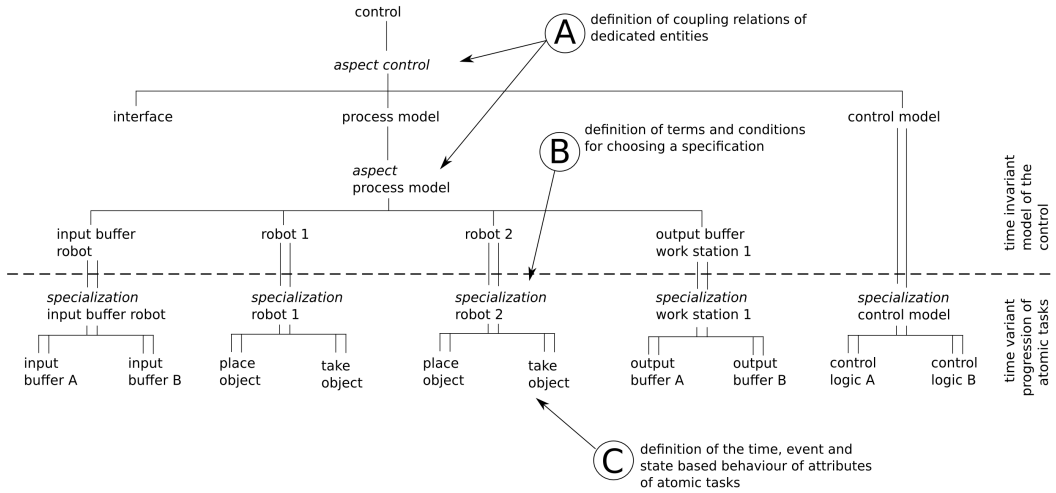

Figure 5: Declarative specification of a cooperative robot control using a control-SES

The leaf nodes (C) of the control-SES define no further decomposable atomic tasks which are implemented as executable components and stored in a model base (MB). The control-SES in figure 5 is incomplete to preserve clarity. Beside nodes and edges a control-SES specifies node attributes. The leaf nodes representing the atomic tasks define the modification of these attributes depending on the real process behavior. These attributes are described in more detail in the next subsection. The aspect nodes (A) define the coupling relations of the succeeding entities. Furthermore specialization nodes (B) define selection rules used for choosing dedicated atomic tasks.

The general structure of all control variants follows from the time invariant upper part of the control-SES. A valid control variant is synthesized from the control-SES using a parameter vector that maps the current process behavior in terms of states and events to attributes of the control-SES. The result of this synthesis is a reduced tree structure where all decision nodes like specialization or multiple-aspect are resolved. Following [3] this procedure is called “pruning”. During the pruning process all entity nodes in the undermost layer of the time invariant part of the control-SES are substituted with leaf nodes of the time variant part which represent atomic tasks. The selection of atomic tasks is made within the specialization nodes by analyzing the actual parameter vector. The structure of two temporary control variants synthesized from the control-SES in figure 5 is schematically shown in figure 6. The control structures pictured in figure 6 represent only a subset of the tasks necessary for a valid temporary control variant according to the described application.

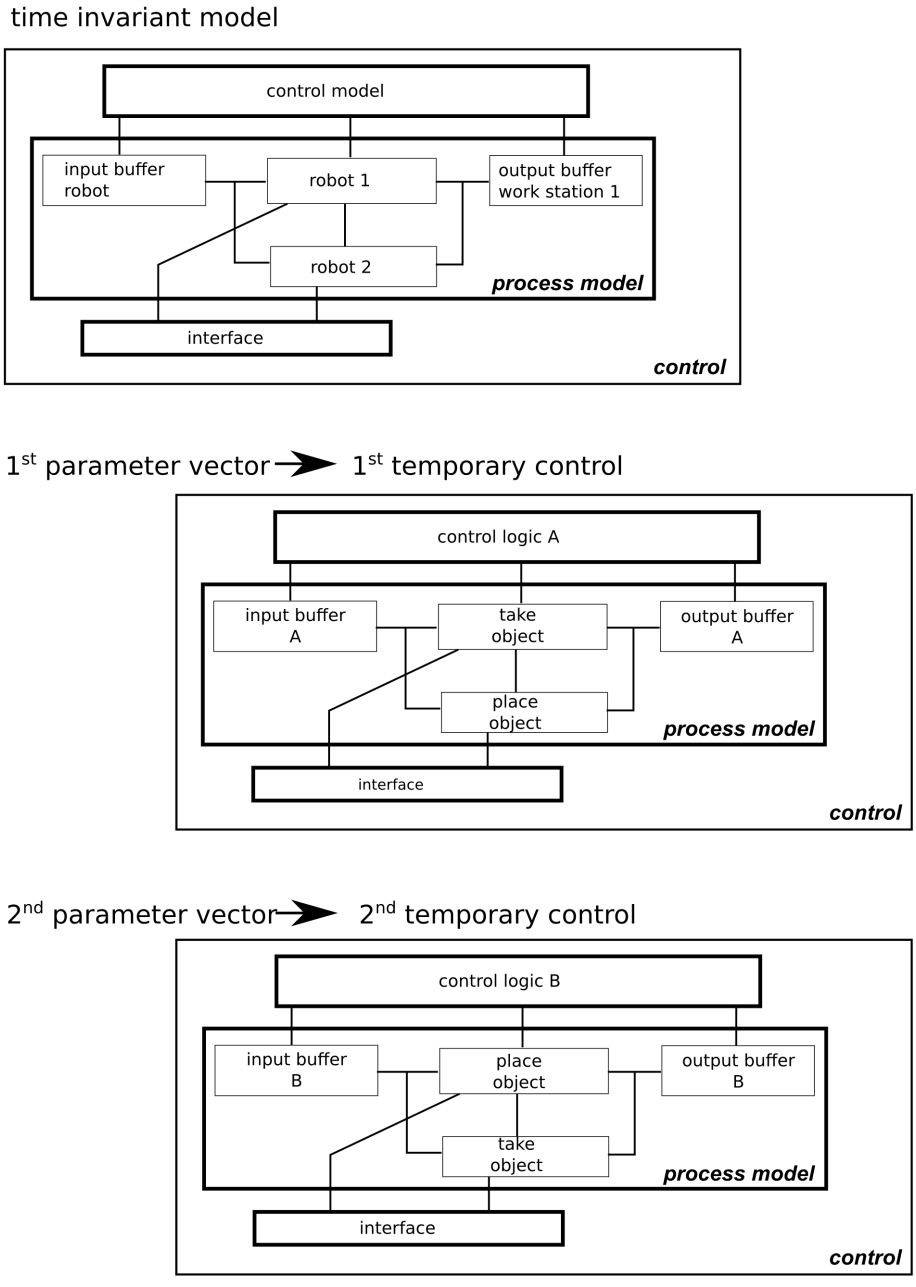

Figure 6: Generation of temporary control variants from a control-SES

3.3 Declarative specification of robot controls

The sequence of atomic tasks is determined during the execution of the control on the basis of current process states and events. The flexibility of the control follows from the

iterative composition and generation of temporary valid control variants. Therefore, a prerequisite is the decomposition of the entire control in appropriate atomic tasks. The specification of atomic tasks and the definition of their sequence will be shown by means of robot 1 of the control-SES in figure 5. For this purpose figure 7 shows in more detail the node *robot 1* and its succeeding nodes.

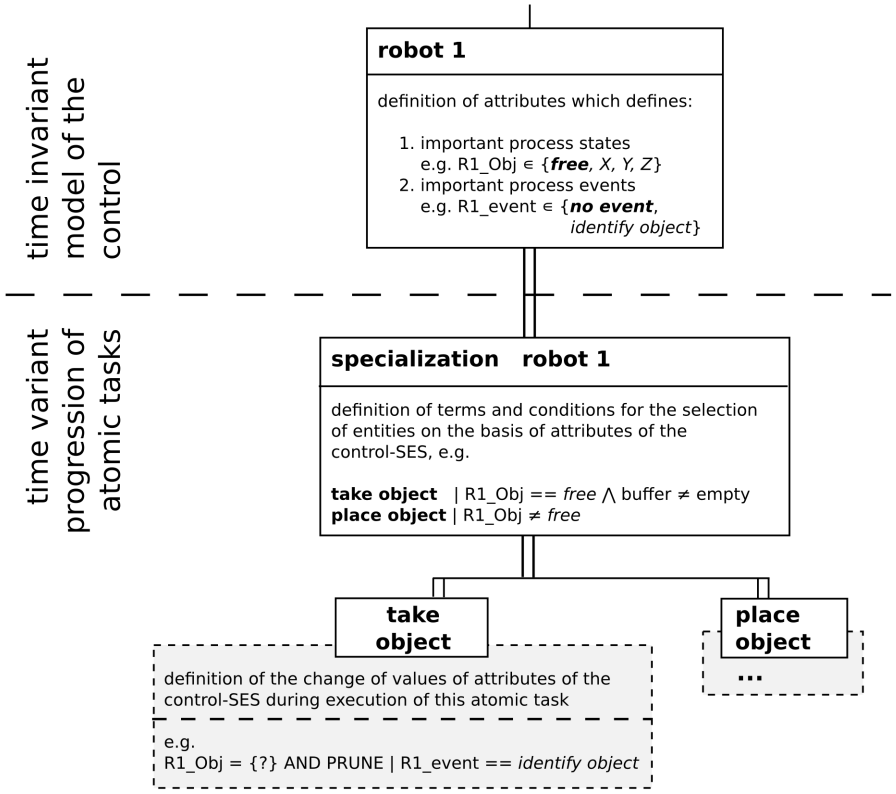

Figure 7: Extended specification of the control-SES for robot 1

Significant states and events including their domain of definition have to be defined with attributes at entity *robot 1*. The succeeding node *specialization robot 1* defines selection rules for the atomic tasks evaluated during the control synthesis process. The atomic task *take object* will be selected during the control synthesis if *robot 1* is currently unused (R1_Obj == free) and if any objects can be taken from the input buffer (buffer ≠ empty). Moreover, every atomic task defines a time, state and event dependable behavior. The task scheduler monitors the attributes of all tasks in a current control and interrupts the control if conditions of attributes are true. Then it updates the parameter vector for a new control synthesis with relevant process states and events and starts a new control synthesis by analyzing the control-SES. When, e.g. the atomic task *take object* is part of the current control the condition R1_event==*identify object* has to be observed. It becomes true when an object has been identified in the input buffer and has been taken by robot 1. Then the object identifier is stored in R1_Obj and the identification event is stored in R1_event of the parameter vector. Moreover, the task scheduler activates a new

control synthesis, because of the PRUNE action coded in the attribute of atomic task *take object*. Now the control-SES is analyzed using the new parameter vector and the atomic task *place object* is selected in the specialization node *robot 1*, because the rule "place object | R1_obj ≠ free" is valid.

4 Automatic generation of control programs

Figure 8 shows schematically the automatic generation of control programs. The starting of any new control synthesis is a parameter vector that contains current process values necessary for the evaluation of attributes defined in the control-SES. At first the task scheduler analyzes the control-SES using the current parameter vector and generates a concrete control variant as parameterizable tree, called Pruned Entity Structure (PES). The PES defines a unique control variant, where the leaf nodes present atomic tasks, which are stored as parameterizable software components in a model base (MB). At second the task scheduler / control generator generates considering the information of the PES and using atomic tasks from the MB an executable control program following the SBC approach and finally brings it into service. During control operation the attribute values of atomic tasks are changing continually according to the real process behavior. This leads to a "PRUNE" action that activates a new control synthesis by the task scheduler / control generator. The cycle has to be repeated until the occurrence of any abort criteria.

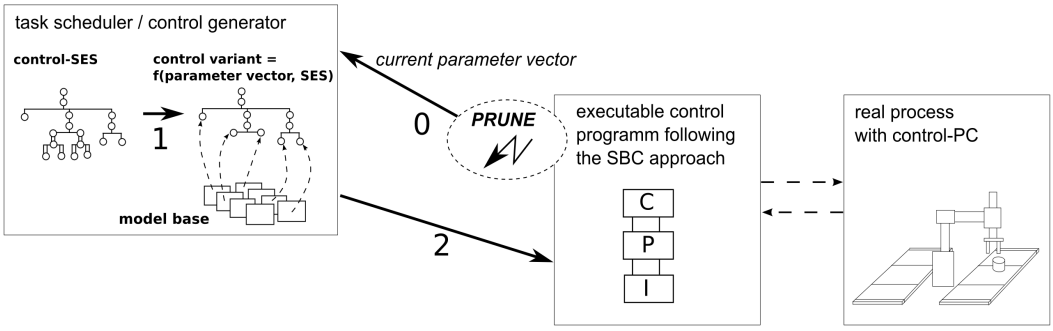

Figure 8: Automatic generation of control programs

Figure 8 illustrates the modularization which is used by the FTC/SES method. The strict separation of control specification and implementation of atomic tasks as parameterizable components supports the development of high flexible controls. Moreover, individual modules of the MB can be extended easily because interactions are only specified within the control-SES. Hence, atomic tasks can be developed in a collaborating team.

5 Summary

The introduced FTC/SES method for a systematic development of flexible task oriented robot controls based on the System Entity Structure / Model Base formalism and the Simulation Based Control approach has been prototypically implemented and tested

in the programming environment MATLAB. The iterative pruning of the control-SES to derive valid control variants is implemented by a MATLAB interface to SWI-Prolog. The atomic tasks stored in a model base have been implemented in MATLAB based on the DEVS formalism [5,6]. Hence, each generated control variant presents a modular hierarchical DEVS model that is executed by a realtime synchronized DEVS simulation environment which is also implemented in MATLAB. The discussed application of a cooperating robot control has been completely implemented using the introduced FTC/SES method. The atomic tasks did not contain any interrelations and they are usable for different robot types, because the interface to the real process is separated. The interface component of the robot application has been implemented using the MatlabKK-Robotic Toolbox [8]. The FTC/SES method simplifies bringing the robot application into service because any maintenance of atomic tasks are focused only on the modules of the model base. Next work will be focused on the development of further applications using the FTC/SES method to prove the approach.

References

[1] Haun, M.:
Handbuch Robotik. (english translation: *Handbook of Robotics*)
Berlin, Heidelberg: Springer Verlag, 2007

[2] Maletzki, M.; Pawletta, T.; Pawletta, S.; Dünow, P.; Lampe, B.:
Simulationsmodellbasiertes Rapid Prototyping von komplexen Robotersteuerungen. (english translation: *Simulation-Based Rapid Prototyping of Complex Robot Controls)*
atp-Automatisierungstechnische Praxis,
Oldenbourg Verlag, München, 50(2008)8, 54 - 60

[3] Zeigler, B. P.; Hammonds, P.E.:
Modeling and Simulation-based Data Engineering.
Burlington, San Diego, London: Elsevier Academic Press, 2007

[4] Abel, D.; Bollig, A.:
Rapid Control Prototyping, Methoden und Anwendungen.
Berlin, Heidelberg: Springer Verlag, 2007

[5] Zeigler, B. P. ; Prähofer, H. ; Kim, T. G. :
Theory of Modeling and Simulation, 2. Aufl.
San Diego, San Francisco, New York, Boston, London, ... : Academic Press 2000

[6] Schwatinski, T.; Pawletta, T.; Pawletta, S.; Kaiser, C. :
Simulation-based development and operation of controls on the basis of the DEVS formalism.
Proceedings of The 7th EUROSIM 2010 Congress, Prag, Czech Republic

[7] Jacak, Witold:
Intelligent robotic systems: design, planning, and control.
New York: Kluwer Academic / Plenum Publishers, 1998

[8] Christern, M.; Schmidt, A.; Schwatinski, T.; Pawletta, T. :
KUKA-KAWASAKI-Robotic Toolbox for Matlab.
Hochschule Wismar,Website, 2011
http://www.mb.hs-wismar.de/cea/KK_Robotic_Tbx/KK_Robotic_Tbx.html