International Journal of Computers, Communications & Control
Vol. I (2006), No. 4, pp. 53-60

An Iterative Method for the Design Process of Mode Handling Model

Nadia Hamani, Nathalie Dangoumau, Etienne Craye

Abstract: This paper focuses on formal verification and validation of a model dedicated to *mode handling* of flexible manufacturing systems. The model is specified using the synchronous formalism Safe State Machines. A structured framework for the design process is presented. The obtained model is characterized by a strong hierarchy and concurrency that is why within the design process an iterative approach for specification, verification and validation is propose in order to improve this process. The main properties being verified are presented and the approach is illustrated through an example of a manufacturing production cell.
Keywords: Flexible Manufacturing Systems, supervision, mode handling, functional and behavioral modeling, verification, validation.

1 Introduction

According to our design approach of fault tolerant control systems, *mode handling* is a function of supervision. In view of a disturbance (failures, breakdowns) *mode handling* allows implementing the decisions about mode and configuration changing. The design of *mode handling* function needs to provide a model representing the operating modes of the production system and its subsystems. To this aim, it is important to use an adequate modeling method and powerful specification formalisms. Those characterizing the most significant approaches are compared in [9]. We proposed in our early work a modeling approach for reactive *mode handling* of Flexible Manufacturing Systems (FMS) [10]. This approach is based on a functional modeling [11] and a synchronous reactive approach using Safe State Machines (SSM) [1] [2].

Due to increasing complexity and flexibility of FMS, some problems characterize mode changing if coherence and safety constraints are not taken into account in the specification / modeling stages. So it is necessary to verify and validate the proposed models at the early stages of the design process. A greater interest has been allocated for few years to formal verification methods, which guarantee that for all possible evolutions of a model, several properties are satisfied. For our modeling approach, one the interests of using SSM formalism relies on its strong semantics leading to the possibility of using analysis tools in order to formally prove that the behavior of the system respects some properties.

The purpose of this paper is to present the iterative approach used in the design process of the model dedicated to *mode handling*. The paper is organized as follows. In section 2 the basic concepts of V&V are reminded and the design process based on formal methods is introduced. The stages of this design process including V&V are detailed in section 3 and section 4. The iterative approach is introduced and the main properties being verified within the design process are presented. Finally the approach is illustrated using an example of a flexible manufacturing cell.

2 The design process

A design process should involve many intermediary stages of V&V. This enables to detect and eliminate as soon as possible the specification or modeling errors. The modifications are thus carried out with a lower cost in particular in the final step. Indeed, V&V characterize any correct development process. According to [14] verification is the proof that the internal semantics of a model is correct independently of the modeled system whereas validation determines if the model corresponds to the

attempts of the designer formulated in the functional requirements. The required properties are either **generic**, they depend on the formalism we use, or **specific** to the modeled system.

V&V consists of analyzing the properties that should be satisfied by the final model using some analysis tools. Two complementary **methods** exist, simulation and formal verification. We are interested in our study only in **formal methods** for V&V. For formal verification, we have to build a formal model representing the behavior of the system and formal specifications of the properties. Fig. 1 shows the design process including V&V:

- The formal specification as well as the structured modeling process that we remind in the next section, allow ensuring a correct development process.
- V&V of the obtained model is an iterative approach characterized by some corrections, which require reconsidering the specification/modeling stages as needed.
- Based on the verified and validated model the implementation is ensured through automatic code generation.

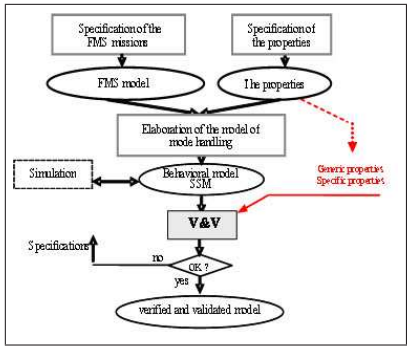

Figure 1: The design process

3 Specification/Modeling

The functional requirements provide the informal specifications about *mode handling* function: the intended behavior of the FMS and the properties that should be satisfied. Other properties such as determinism are also mandatory. The designer formalizes those informal requirements to provide **formal specifications** of the model, of the required properties and, if necessary, some assumptions about the environment. The **formalization** task provides the following models.

3.1 The formal model representing the behavior of the system

The specifications of the model dedicated to FMS *mode handling* are detailed in our earlier work [10][11]. We remind in the following the main characteristics of the modeling process.

The FMS subsystems. In our approach, an operating sequence characterizes each part. The specification method of FMS subsystems was described in [11]. The idea is to decompose and identify the functional subsystems, which take part in the realization of its missions. A mission corresponds to a set of operating sequences that the system should produce simultaneously. The FMS functional model is obtained by a hierarchical decomposition leading to the elementary machining and transfer operations that the FMS performs. The structural aspect completes this specification by associating the resources to the elementary operations they perform. The obtained subsystems are organized in six layers as shown in Fig. 2.

The functional representation is a graph composed of functional subsystems (called entities) linked by logical relationships (AND, OR). OR nodes correspond either to an inclusive or an exclusive logical operator according to the constraints given in the functional requirements.

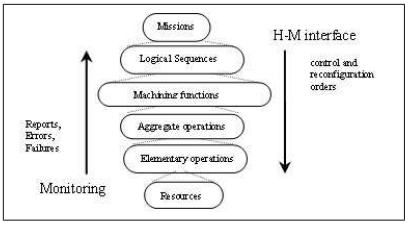

Figure 2: The FMS functional model

Behavior specification. Given the FMS functional model, one should provide behavioral specifications. It is necessary to represent the activation and deactivation mechanisms at all the hierarchy levels of the specification as well as the availability of the resources and the functions of the FMS. The activation/deactivation of an entity is represented by the working mode. The functioning mode represents the availability/unavailability of an entity. The aim is to handle concurrently the information flows downwards to transmit high-level control and reconfiguration orders and upwards to follow up the reports and failures detected by the monitoring function. The reactivity needed for this bi-directional exchange of information, in addition with the characteristics of concurrency and preemption, require a synchronous approach. The behavior of the identified FMS subsystems and the logical relationships between them are specified using the synchronous formalism SSM [1][2]. SSM inherits many features from Statecharts [12] but offers several forms of preemption and benefits of more strict semantics fully compatible with that of Esterel synchronous language [4]. SSM supports also, with a very rigorous semantics, hierarchy, communication, concurrency, and various forms of preemption, which characterize our modeling approach. In addition, SSM takes advantage of an industrial development environment [7], which provides necessary tools for a design process.

The behavior of each entity of the model, whatever its level, is represented by a SSM model. This model allows knowing, at any time, if an entity belonging to the current mode of the FMS is in normal state *e_N*, degraded state *e_D* or out of order state *e_OUT* according to the functioning mode point of view; and if it is activated *x_Active* or inactive (in stopping state) *e_OFF* according to the working mode point of view.

The change-of-state of the entities of the model belonging to successive levels is carried out according to precise rules which depend on the kind of the logical relation and the number of child entities. Several cases are studied in [10].

For the representation of the model, we proposed some reference models specified using SSM [10]. These are generic models depending on the number of the child entities and the logical relationships that connect them. The reference models are instantiated for each entity of the model. The proposed method is iterative, it is necessary to specify the models of level *i* , then we encapsulate the SSM of level *i* in the SSM of level *i+1* until completing the specification of the model (see Fig. 3). The encapsulation of the models enables representing the hierarchical levels which characterize the model. The aim is to improve the legibility in the specification/modeling stage. The hierarchy is represented using the encapsulation of successive levels whereas the entities belonging to the same level are separated by dashed lines; these are used to represent concurrency in SSM syntax (see [2]).

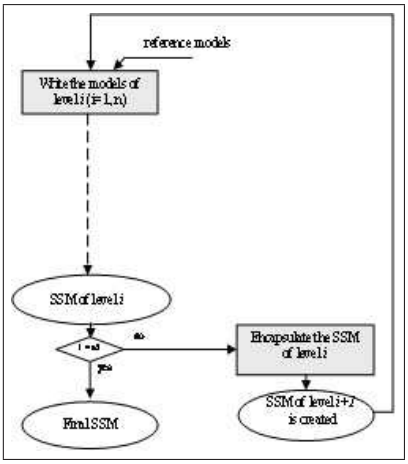

Figure 3: The modular specification

3.2 The formal specification of properties

Predictability and dependability are the main characteristics of reactive systems. In our work, we consider only properties of logical behavior such as *determinism* and *safety properties*. As we adopt a reactive synchronous approach based on SSM formalism, **generic properties** of reactivity and determinism should be verified. The rigorous semantics of SSM enable to provide formal verification mechanisms ensuring such properties. Several **properties specific** to *mode handling* function can be verified. The main properties are presented in section 4.

4 Verification and validation

In the verification stage (Fig. 1), one checks if the model satisfies **generic properties**. SSM specifications are automatically translated into Esterel program which is compiled in a system of Boolean equations (logical circuits) an implicit form of Finite State Machines (FSMs). The properties of logical correctness of Esterel programs can then be verified on this system. The **analysis technique** used to this aim is the constructive causality analysis carried out by Esterel compiler v5_x. Causality analysis is a semantic analysis which allows accepting or rejecting Esterel programs according to whether they are constructive or not [5]. The processor which implements the algorithms of constructive analysis builds the reachable state space using symbolic techniques based on Binary Decision Diagrams (BDD).

One checks in the validation stage if the model satisfies the **specific properties** formalized in the specification stage. The **analysis techniques** used are *model-checking* and/or *theorem proving*. Several *model-checking* tools were developed for Esterel language. For instance *Built-in verifier* tool an evolution of XEVE *(XEsterel VErifier)* [6] is currently integrated into Esterel Studio. XEVE takes as input the system of Boolean equations and built the reachable state space using BDD (symbolic *model-checking* [13]) in order to check the statutes of output signals of the model or the observers [8] representing the properties.

Failing of V&V stage implies that there is a specification or a modeling error. It is then necessary to go back over this model as needed. The formal specifications are may be incorrect. The errors are either in the specifications of: the model, the property, the correspondence between the model and the property or the assumptions related to the environment.

If the specification/modeling stages are considered to be correct then we should reconsider informal requirements. Indeed, within the formal specification of the model or the properties, errors are may be caused by the ambiguity or the incompleteness of the informal requirements provided in the functional

requirements. In this case, these requirements must be improved and formalized again.

We propose to complete the modular and hierarchical specification shown in Fig. 3 by introducing an iterative method of V&V. This method is presented in the following. The aim is to reduce the specification/modeling errors and to enable their early correction by introducing intermediate stages of V&V into the multi level specification stage.

4.1 An iterative method for V&V

We propose at first to check each single reference model used for the specification of the model. The aim is to prove reactivity and determinism. The verification of these reference models is performed using the analysis tools integrated into Esterel Studio environment.

The single verification of the reference models:
For each reference model
Write the model
Verify the model
If the verification fails, correct the specification and repeat the verification as needed.

The single reference models checked in this way are instantiated in order to obtain the behavioral models of the FMS subsystems. Then the integration of these models allows editing progressively the whole model. Using the SSM graphical editor, the best way is to follow a bottom up approach based on the encapsulation of SSM of level *i* in the SSM of level *i+1*. At each level we should be sure that the intermediary SSM satisfies some properties and the accumulation of errors will be avoided. That is why we propose to associate with this specification an iterative method of V&V. If the V&V does not converge then it is necessary to correct the specification and to repeat the V&V process as needed. Indeed, V&V is carried out in an iterative way until the whole model (the final SSM) will be verified and validated. The method described in the following is represented in Fig. 4.

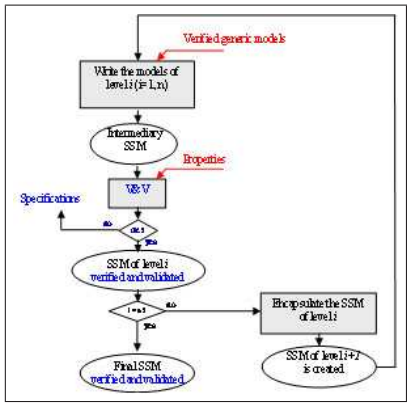

Figure 4: The iterative V&V

Repeat for all the models of level i (i = 0, n)
(1) Write the models of level i
The specification is carried out by instantiation of the predefined and verified models
(2) Verify then validate the SSM of level i
If the V&V fails, correct the specification and repeat the previous step as needed
If i = n go to (4) if not
(3) Encapsulate the models of level i 'SSM of level i+1 is then created'
i = i + 1
End of the procedure

(4) The SSM represents the FMS model (the top level corresponds to the whole model)
If the previous steps converge, we conclude that the final SSM is verified and validated. When the final SSM is verified and validated, the code could be generated.

The properties being verified within this process are presented in the following.

4.2 The properties

The model dedicated to *mode handling* is translated automatically into Esterel language. Esterel compiler is composed of several processors for type checking, syntactical and lexical analysis. Semantic analysis also called causality analysis allows checking **generic properties** (deadlock freeness, reactivity, determinism).

Deadlock freeness. Causality analysis makes it possible to prove that the program is causal according to the constructive causality defined by Berry [5]. Constructive causality of Esterel programs ensures deadlock freeness, which characterizes synchronous languages. Deadlocks are due to the synchronism assumption (see causality problems in [3]).

Reactivity and determinism. If an Esterel program is constructive then it is reactive and deterministic. A program is said to be reactive if it provides a well-defined solution for each input. It is deterministic if this solution is unique [5].

The aim of **specific properties** is to ensure coherence and safety constraints of *mode handling*. We have proposed three kinds of safety properties: mode reachability, the mission uniqueness of a FMS and mutual exclusion of incompatible modes.

States reachability. We can check the reachability of the states specified by the model (according to the specifications, an output signal is associated with each control state of the model). To this aim, the statutes of output signals that correspond to control states of the model are tested.

The mission uniqueness of a FMS. This property ensures operation safety of the FMS according to the selected mission. Indeed, selecting a mission implies performing some operations and activating the resources, which take part in this mission. Thus, the operations and the resources, which do not take part in the current mission, should not be activated for safety and coherence reasons. The property is guaranteed by construction of the model.

Incompatibility of the modes. In order to ensure mode coherence, the proposed specifications guarantee mutual exclusion of incompatible states (each state is belonging to a distinct mode) of the entities of the model. It is about mutual exclusion of working states of the entities related with the logical relation XOR (for example redundant resources).

The previous properties are verified within the iterative approach of V&V. The intermediate verifications are important because if causality cycles are detected early their correction will be easier. Contrary to verification, intermediate validations are may be expensive and some properties are checked only on the final model, in particular the properties related to the uniqueness of the FMS mission.

In addition to the main properties, other properties of *mode handling* model can be checked. They deal with the conditions of mode changing and the respect of modes sequence [10].

5 Illustration

We have applied our approach to the example of the cell represented in Fig. 5. This cell consists of two machines M1, M2 and input/output buffers (I/O). The transport system is composed of two robots R1, R2. We suppose that M1 is loaded by R1 and M2 is loaded by R2, this later will be used to load the

two machines when R1 is failing. The machining functions performed by this system are turning and milling. Turning (t) is carried out by M1, milling (f) by M1 or M2. The cell has three missions, *M1*, *M2* and *M3*. The corresponding operating sequences are respectively OS1 (t), OS2 (f) and then OS1 and OS2 simultaneously.

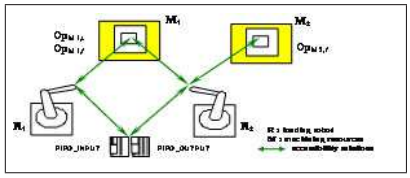

Figure 5: Illustration example

The obtained model using our modeling method is specified using 4 reference models and 29 instances for 809 lines of Esterel code generated from the SSM models. This model needs 34 input signals and 145 output signals.

We simulated at first some scenarios of behavior in order to correct the specification errors so that the intended behavior will be in conformity with the functional requirements. The interactive simulator XES [7] is used.

The studied cell can be extended with addition of machining resources (simple or polyvalent) and redundancies of the transport system (robots, conveyor). This allows studying the problem with increasing complexity. For the specification, adding or removing reference or instantiated models enable integrating easily the changes on the model thanks to the modularity and the hierarchy of the specification approach. However, due to increasing complexity of the cell some tests performed by the model-checker could not be concluded (out of memory problem). We try to solve this problem in a future work.

6 Summary and Conclusions

This paper deals with V&V within the design process of a model dedicated to *mode handling* of FMS. The first contribution of this study is a structured process for carrying out V&V. It is an extension of our control system design approach in order to take into account V&V stage. An iterative method is used for specification and V&V stages of the design process. The aim is to refine the model construction by introducing intermediate steps of V&V for an early correction of errors. Within the proposed framework the required properties (determinism, reachability, mutual exclusion of incompatible modes,..) are given. The analysis tools integrated into Esterel Studio are used in this process.

The proposed approach can be extended to deal with environment constraints in the specification stage. In addition, further work aims at studying some examples of increasing complexity.

References

[1] C. André, Representation and Analysis of Reactive Behaviors: A Synchronous Approach, in Proc. of the IEEE-SMC Computational Engineering in Systems Applications (CESA 96), Lille, France, July, 1996, pp. 19-29.

[2] C. André, Semantics of SSM (Safe State Machines), Guyancourt, France, April 2003, Available at: http://www.esterel-technologies.com

[3] A. Benveniste, P. Caspi, S. Edwards, N. Halbwachs, P. Le Guernic, and R. de Simone, The Synchronous Languages Twelve Years Later, in Proc. of the IEEE, 91(1), special issue on Embedded Systems, pp. 64-83, 2003.

[4] G. Berry, G. Gonthier, The Esterel synchronous programming language: design, semantics, implementation, *Science of Computer Programming*, Vol. 19(2), pp. 87-152, 1992.

[5] G. Berry, *The constructive semantics of pure Esterel* Draft version 3, 1999, Available at: http://www.esterel-technologies.com

[6] A. Bouali, XEVE an Esterel Verification Environment, in Proc. 14th Int. Conf. on Computer Aided Verification CAV'98, LNCS, UBC, Vancouver, Canada, June 1998.

[7] Esterel StudioTM, Available at: http://www.esterel-technologies.com.

[8] N. Halbawachs, F. Lagnier and P. Raymond, Synchronous observers and the verification of reactive systems, in Proc. of AMAST'93, 1993.

[9] Hamani N, Dangoumau N, Craye E, A comparative study of mode handling approaches, in Proc. of the 35th International Conference on Computers & Industrial Engineering (CIE 05), Istanbul Turkey, June 19-22, 2005.

[10] N. Hamani, A contribution to modeling and verification for mode handling of manufacturing systems, PhD dissertation, Ecole Centrale de Lille, France, 2005 [in french].

[11] N. Hamani, N. Dangoumau, E. Craye, Functional modeling for mode handling of Flexible Manufacturing Systems, in Proc. 12th IFAC Symposium on Information Control Problems in Manufacturing, Saint Etienne, France, 2006.

[12] D. Harel, StateCharts: a visual approach for complex systems, *Science of Computer Programming*, Vol. 8, nř3, pp. 231-275, 1987.

[13] K.L. McMillan, *Symbolic Model Checking*, Kluwer Academic Publishers, 1993.

[14] J.M. Roussel, and J.J. Lesage, Validation and Verification of Grafcet using finite state machines, in Proc. of IMACS-IEEE Multiconference on Computational Engineering in Systems Applications, Lille, France, 1996, pp. 758-764.

Nadia Hamani, Nathalie Dangoumau, Etienne Craye
Laboratoire d'Automatique, Génie Informatique et Signal (LAGIS), CNRS UMR 8146
Ecole Centrale de Lille, BP. 48, Villeneuve d'Ascq, F-59650 France
E-mail: nadia.hamani@ec-lille.fr

Received: November 6, 2006