Modular and Multi-view Representation of Complex Hybrid Systems

Luc THEVENON, Jean-Marie FLAUS
Laboratoire d'Automatique de Grenoble
UMR 5528 – CNRS-INPG-UJF
ENSIEG, BP 46, F-38402, Saint Martin d'Hères Cedex
FRANCE

Abstract: - A modular and multi-view approach for the modelling of complex hybrid systems is presented. The aim is to define a global representation of this class of system, i.e. a representation which groups together in a same structure several types of information coming from different origins. Such a model, used for different purposes like the simulation of the process or the diagnosis, constitutes a first step toward an objective of supervision of the plant. For that a decomposition of the process based on the notion of input-output blocks is examined, and a hybrid generic object is then deduced. The modular building of the process model from these objects is shown, as the way to project the model on different views.

Key-Words: - Hybrid systems, complex systems, batch plant, multi-models approach, modular modelling, supervision *IMACS/IEEE CSCC'99 Proceedings*, Pages:2491-2496

1 Introduction

Our aim is to define a modelling approach for hybrid complex systems which is based on the idea that any complex system can be analysed with a *systemic* point of view and seen as a set of basic elementary systems interacting together. Our challenge is to find the right standard atomic system, to define its structure in order to be able to describe any behaviour, and then to define a semantic for the interactions which is rich enough in order to allow the description of complex systems. As a by-product of our approach, we propose a formal definition of a multi-view modelling approach in which the main idea is to project (or restrict) the complete system on a set of a relation of a specific type.

We will illustrate our idea on a class of such systems which are batch processes. Indeed, these processes are becoming more and more important in industries dealing with material transformation (chemical, food biotechnological or pharmaceutical industries) thanks to the high level of flexibility of these processes which allows to manufactured several types of products in small quantities with the same equipment. A model of such complex systems must hold enough information for different purposes, like the simulation of the process or the diagnosis, etc. It constitutes a first step toward an objective of *supervision* of the plant by offering a method to represent with a coherent way information coming from different origins.

The idea is to represent the plant according different points of view (physical, behavioural, functional, etc.), and to mix these views to get a full model of the plant. We must then define a structure in order to have an internal unified way of representing an object with their different views and the rules to combine these views. The structure must allow to decompose the whole process in smaller parts easier to manipulate in order to reduce the complexity of the modelling. The main characteristics of the representation are the following:

- **global** in order to take into account all the various parts and the main aspects of the plant (topology, physical transformation, control, etc.)
- **modular** in order to build the whole model of the plant (often complex) by interconnection and aggregation of basic objects,
- **flow transformation oriented**: an object is a physical equipment like a valve or a reactor which acts on the energy or material flow, in order to build the plant model by following its topology, and to reuse objects already defined.

For that, we have defined a representation based on the notion of input-output blocks interconnected. Block diagrams used in Simulink for the modelling of continuous systems are a classic example of such an approach, whose the interest lies in the modularity of the model building. We have extended this approach to the hybrid dynamical systems and

defined a generic input-output object (section 2). Such an attempt has been made in [1] and [2] where the continuous dynamic is approximated by a discrete model, and in [3] and [4]. Then, we show the building rules of complex objects in section 3, and the projection of the full model of the process on different views in section 4.

2 Generic Granular Object

We have defined a generic object with an internal dynamic which allows to represent any physical equipment belonging to a complex hybrid process. This dynamic is described by a continuous subsystem and a discrete one which are interconnected by their inputs-outputs through appropriate interfaces. But, in order to describe the behaviour of a hybrid process, continuous signals are not sufficient, and other input-output types must be defined. We present them in this section, and then we describe the internal dynamic of the generic object.

2.1 Input-output signals and flows

First, let us define the notion of *variable* and *vector*.

Definition 1: Variable

There are four types of variables.

- ***Continuous variables*** *are described by a real or integer number and can take an infinite number of values.*
- ***Boolean variables*** *have two possible values:* TRUE *or* FALSE.
- ***Symbolic variables*** *(noted between "") are described by a character string ("open", "start") and can take a finite number of values defined in a list.*

Definition 2: Vector

A vector is a n-uplet composed of variables of the same types (simple vector) or of different types (composite vector).

The inputs-outputs of a generic object can be filed in two categories: *signals* which convey information, and *flows* which describe a physical flow. Contrary to signals for which the information is always conveyed in the same direction (actuator towards process, sensor towards controller, etc.), the direction of a flow can change during the evolution of the system, according to the direction of the physical flows of the process. Signals and flows can be unitary or multiple (composed of several signals or flows).

2.1.1 Continuous signals ()

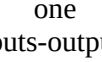
They convey continuous information (actuator or sensor values). A continuous signal is represented by a continuous variable.

2.1.2 Event signals ()

They convey discrete state transition information of a system. An event signal is a pointwise non-zero signal represented by a Boolean variable.

2.1.3 Condition signals ()

They convey information about the discrete state of a system. A condition signal is a piecewise constant signal represented by a symbolic variable.

2.1.4 Continuous flows ()

They describe the physical flows which circulate into the system at a given point. They are specific to an application domain. For example, in the case of chemical engineering, they describe the material and energy flow. A continuous flow is represented by a continuous vector and contains the following information about the physical flow:

- *Rate of flow*, i.e. a continuous variable.
- *State*, i.e. extensive and intensive continuous variables (pressure, concentrations, etc.).
- *Characteristics*, i.e. a set of constant parameters (density, etc.).
- *Direction*, i.e. an information concerning the current direction of the physical flow.

2.1.5 Discrete flows ()

They describe the flows of physical objects which circulate into the system at a given point (bottles or barrels). Contrary to the continuous flow for which several components can be present at a given point of the flow (mixture of two liquids), only one object is present at a given point of a discrete flow (a bottle). A discrete flow is represented by a composite vector and contains the following information about the flow of physical objects:

- *Rate of flow*, i.e. a Boolean variable which is TRUE when an object is present as regards of the flow, and FALSE the rest of the time.
- *Characteristics*, i.e. a set of constant parameters which describe the current object (size, color, composition ...).
- *Direction*, i.e. an information concerning the current direction of the flow of physical objects.

2.2 Continuous subsystem

It represents the continuous dynamic of the generic object. Many formalisms exist to describe such a dynamic: transfer functions, bond-graphs, ordinary

differential equations, etc. The more general representation is given by differential and algebraic equations (DAEs) of the form:

$$\begin{aligned} 0 &= f\left(\dot{x}(t), x(t), u(t), t\right) \\ y(t) &= h\left(x(t), u(t), t\right) \end{aligned} \qquad (1)$$

where *x(t)* is the continuous state, *u(t)* is the input vector (set of continuous signals), and *y(t)* is the output vector (set of continuous signals). We add also a set of continuous flows *φ(t)* as input and/or output of the system. But, such a representation does not allow to represent discrete switchings of the system's dynamics. For that, an other input is required to know the discrete state *q(t)*, and to select the appropriate set of equations. As we saw, the discrete state of a system is conveyed by a condition signal. We obtain then the following subsystem:

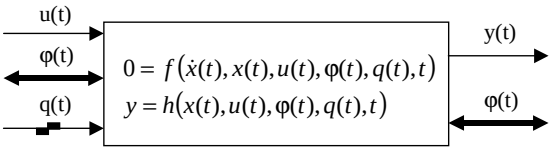


Figure 1: Continuous subsystem

2.3 Discrete subsystem

It represents the discrete dynamic of the generic object. As for the continuous one, many formalisms exist to describe such a dynamic: automaton, Petri nets, Statechart, Grafcet. The weakness of most of them is that the communication between subsystems is not well suited, what not allows to build easily modular models. An other formalism, called condition/event (C/E) systems and introduced by Sreenivas and Krogh [5], is well suited for modular modelling of discrete event systems, and is similar to continuous block diagram. It is based on two types of inputs-outputs: *condition* and *event* signals. As we saw, condition signals allow to convey information about the state of a discrete system. This is very convenient by comparison to automaton which communicate only with event signals, what impose the addition of discrete states in order to obtain the same result. The behaviour of this system is described by three equations, as shown *Figure 2*.

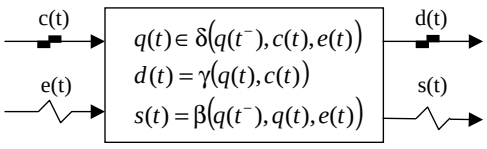


Figure 2: Discrete subsystem

It remains the treatment of discrete flows, for which an other discrete formalism needs to be used to describe their interaction.

2.4 Subsystems interconnection

To interconnect discrete and continuous subsystems, we have defined appropriate signal/flow (S/F) converters.

2.4.1 Continuous S/F to C/E signals converter

Also called *event detection*, it converts continuous signals and/or flows into condition and event signals. When the continuous inputs cross a threshold described by an *event function g* of the form:

$$0 = g\left(w(t), t\right) \qquad (2)$$

an output event signal is generated, and an output condition signal is set to a specific value. Here, the continuous vector *w(t)* is composed of the outputs continuous S/F *y(t)* and *φ(t)*. As input, there are also a condition signal *c(t)*, and an event signal *r(t)* whose the meaning depends on the size of *w(t)*.

If the size of *w(t)* is non-zero, the time of the event depends on the evolution of the continuous inputs (*state event detection*). The input *r(t)* is not used, and *c(t)* conveys the current value of the discrete state in order to select the appropriate set of event functions and remove the ones not yet or not longer useful. Two sets of outputs are considered, according whether the threshold is crossed from a negative value (positive direction), or a positive one (negative direction) as shown Figure 3.

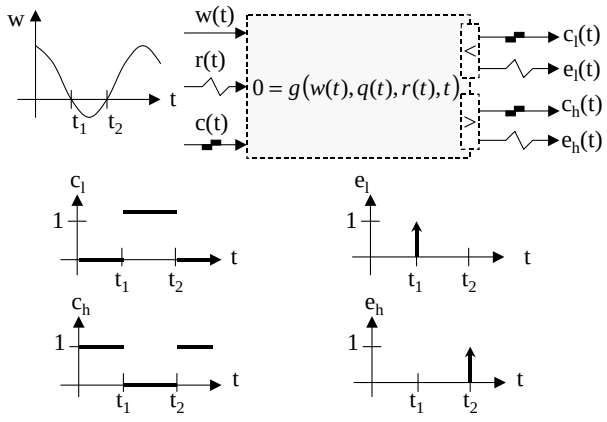


Figure 3: State event detection

In a simulation objective, the exact determination of the time of the state event is very important, but poses many problems, because it can occur during an integration step. This is not the purpose of this paper to explain these problems, but they must be taken in account for such an objective.

If the size of *w(t)* is zero, the time of the event is known, because it depends only of the time *t* (*time event detection*). The converter is then a simple timer with one set of outputs, and whose the start

time is indicated by the input *r(t)* under the condition *c(t)*. When the wanted time is reached, the timer is automatically deactivated until the occurrence of a new input event.

2.4.2 Event signals to continuous S/F converter

It converts event signals into continuous signals and/or flows. When an input event *ev(t)* occurs at t_{ev}, values of a set of continuous variables *z(t)* are modified according to a function f_x (Figure 4). The vector *z(t)* is composed of the input continuous S/F *u(t)* and *φ(t)* what corresponds to a modification of the continuous evolution, and also of the continuous state *x(t)*, what corresponds to a state jump.

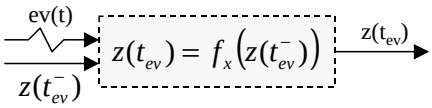


Figure 4: Event to continuous converter

2.5 The generic object

From the discrete and continuous subsystems interconnected through signals and flows converters, a generic hybrid object with the five types of inputs-outputs seen previously can be defined (Figure 5).

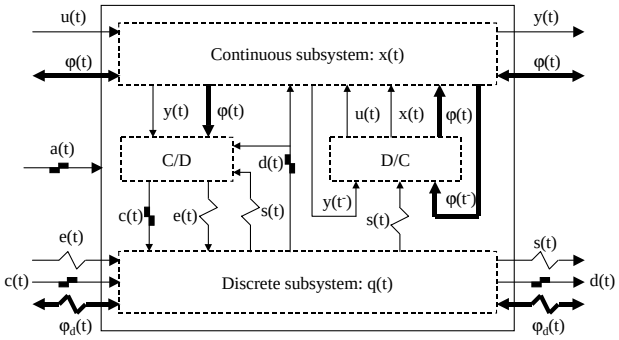


Figure 5: Generic hybrid object

More formally, a generic object is defined by:

- a continuous state vector $x(t) \in \Re^n$,
- a discrete state vector $q(t) \in Q$ a set of symbols,
- a composite input vector (signals and flows),
- a composite output vector (signals and flows),
- a composite input/output vector (continuous and discrete flows),
- a set of continuous equations,
- a set of equation defining the c/e system,
- a super-object and several sub-objects.

We see that flows can be either input or output or input/output of the object, according whether the direction of the corresponding physical flow can change or if it is set by the topology of the process. We show Figure 6 a simple example of the modelling of an on/off valve.

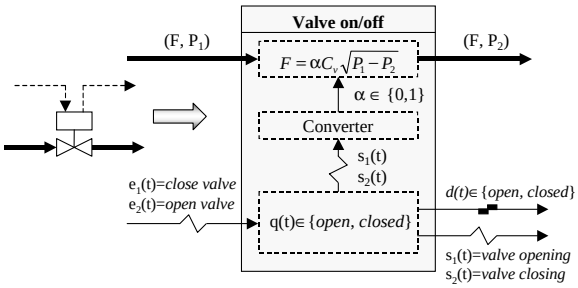


Figure 6: On/off valve object

In a first time, this object can be used in order to represent any basic physical equipment, like for example a valve, a reactor, or a continuous controller. Then, elementary objects can be interconnected by their inputs-outputs in order to build more complex equipment (distillation column), or a sequential control (manufacturing recipe), or the whole model of the plant.

An advantage of such an approach, in addition to the simplification of the model building, is that the internal dynamic of objects can be described with other formalisms (transfer functions, black box model, etc.), or several accuracy levels (static relations). Only the input-output types of the object must be identical to the ones defined previously.

3 Complex objects

Complex objects are built by *aggregation* of other objects, i.e. the new object (called super-object) is composed of other objects (called sub-objects) which can be *interconnected* or not as seen Figure 7. For that, specific rules allow to check the validity of the model and create it.

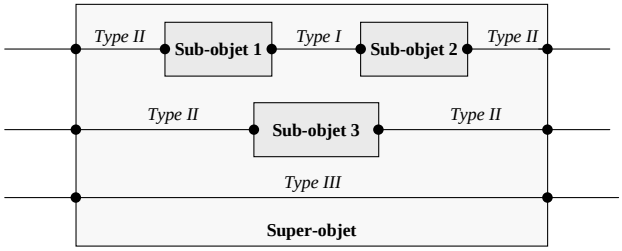


Figure 7: Aggregation of three objects

3.1 Connections between objects

Figure 7 shows the three types of connection between objects, which are realised through variables forming signals and flows.

***Rule 1**: The type and the dimension of the two connected signals or flows must be the same.*

Type I is a connection between two elementary or complex sub-objects. It consists in replacing in the equations of each sub-object the input and output

variables by the corresponding flow or signal variables which become state variables added to the continuous state vector of the super-object.

Type II is a connection between a sub-object and a super-object. It defines the inputs and outputs of the super object from those of the sub-objects.

Type III is a connection between the inputs and outputs of the super-object. The corresponding variables are not modified by the object.

The validity of connections can be checked by looking at the causality of the variables involving in the connection (Input, Output or Input/Output):

Rule 2*: The validity of a connection between two variables A (sub- or super-objet) and B (sub-object) is defined by the following table, according to the type of the connection.* ✓ *means a valid connection, whereas* ✗ *means a wrong one.*

A	*B*	*Type I*	*Type II*	*Type III*
I	*I*	✗	✓	✗
	O	✓	✗	✓
	I/O	✓	✓	✓
O	*I*	✓	✗	✓
	O	✗	✓	✗
	I/O	✓	✓	✓
I/O	*I*	✓	✓	✓
	O	✓	✓	✓
	I/O	✓	✓	✓

3.2 Object aggregation

A complex object has the same structure that a elementary one. Its attributes are defined at the time of the sub-objects aggregation by the following rules:

- The inputs-outputs are defined by those of the sub-objects through type II or III connections. Non connected sub-objects inputs-outputs become non accessible from the exterior.
- Continuous state vector is defined by aggregating those of sub-objects. The inputs and outputs variables interconnected of sub-objects become new states added to the state vector.
- Discrete state vector is defined by aggregating the vectors of sub-objects.
- Continuous system is defined by grouping together the sets of equations of each sub-objects modified by the connections.
- C/e system is defined by the c/e systems of each sub-objects which communicate through input and output condition and event signals which become internal signals for the new object.

3.3 Structural analysis

According to the goal of the modelling, that can need a numerical simulation of the continuous equation system. But, the raw set of equations can not be used without a structural analysis whose the purpose is to check the validity of the continuous system, and to arrange the order of equations.

3.3.1 Structural singularities

During the aggregation of objects, specific rules are applied, and connections are checked. But, nothing assures that the set of continuous equations obtained for the whole system is correct, i.e. that the equation system is *resolvable* in a mathematical or simulation sense. An equation system is resolvable if the number of equation and continuous variables are the same, and if each variable can be paired with an equation in which it appears. If not, the system is said structurally singular and can not be resolved. Because the set of equations picked changes according to the discrete state, such a verification must be performed for all the possible continuous models. One solution to detect such anomalies, is to permute the equations to make all diagonal elements of the incidence matrix non-zero [7].

3.3.2 Causal sets

We define a *causal set S*, as a set of continuous equations (and by extension a set of objects) whose all the input variables are set by the external environment and do not depend on the evolution of *S*, and all the outputs variables are set by *S* and do not depend on the external environment.

The decomposition of a model in several causal subsets or blocks is useful because each block can be simulated independently from the others by using a sequential resolution method, what reduce the size and the complexity of the problem in a numerical point of view. The determination of causal sets can be performed by making a Block Lower Triangular partition of the incidence matrix of the set of equations [7].

4 Notions of multi-view model

A multi-view model is a model which groups together in a same structure, different descriptions or aspects of a process [6]. For example, a batch process can be regards from:

- the equipments which compose it,
- the material and energy transformations that occur into these equipments,
- the automatic systems which control the plant,

- the operating mode which describes how make a product,
- the safety actions realised in case of an abnormal behaviour.

These aspects are described by different types of information (PID diagram, process diagram, automata language, etc.).

In the representation we have defined in the previous sections, each input-output type of an object represents a particular view of this object. So, the full model of a process, represented by all the inputs-outputs, can be projected on different views by considering only the corresponding inputs-outputs. For examples, in a batch process, the transformation view is given by the continuous and discrete flows, the control view by the continuous and C/E signals, and the recipe view by only C/E signals. Sub-views can also be considered like for example the component view which is given by only the continuous flow with the particular component whose the follow-up through the process is wanted.

Such a decomposition reduces the complexity of the model which can be regarded from different views separately, and allows to consider and study only a particular view according to the objectives. Examples of the transformation and control view of a small batch process are shown Figure 8 and 9 where the objects R_i and V_i are the same objects, but projected on different views.

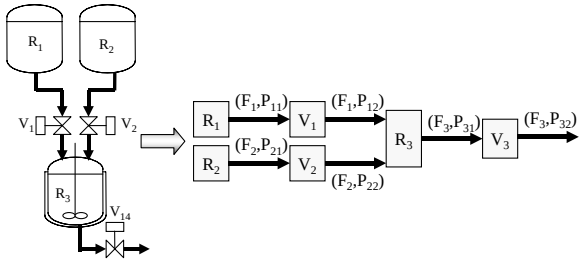


Figure 8: Transformation view

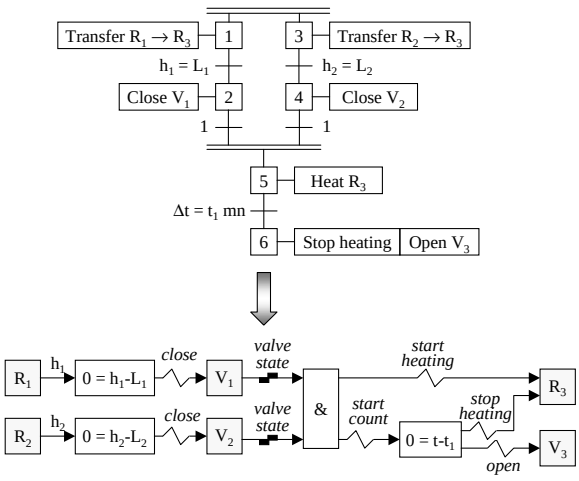


Figure 9: Control view

5 Conclusion

In this paper, we have presented a modular approach for the modelling of complex hybrid systems. The main interests of this approach are:

- to separate the representation (or modelling) phase from the analysis and simulation phase,
- to propose a generic granular object which can be used for modelling complex systems,
- to propose a consistent way of describing relations between the objects,
- to formalise the multi-view notion of a system.

Although we have illustrated it with examples coming from the chemical engineering area, it can be used for the modelling of any continuous, discrete or hybrid complex process. Then, the model obtained holds enough information for different purposes (simulation, etc.), and can be used in an objective of supervision of the plant.

References:

[1] S. Kowalewski, S. Engell, M. Fritz, R. Gesthuisen, G. Regner and M. Stobbe, Modular Discrete Modelling of Batch Process by Means of Condition/Event Systems, *Workshop Analysis and Design of Event-Driven Operations in process Systems*, 1995.

[2] O. Stursberg, S. Kowalewski, J. Preuβig and H. Treseler, Block-diagram based Modelling and Analysis of Hybrid Processes under Discrete Control, *3rd International Conference on Automation of Mixed Processes: Hybrid Dynamical Systems*, 1998, pp. 63-70.

[3] B.H. Krogh, Condition/Event Signal Interfaces for Block Diagram Modeling and Analysis of Hybrid Systems, *Proceedings of the 8th International Symposium on Intelligent Control Systems*, 1993, pp. 180-185.

[4] S. Engell, Modelling and Analysis of Hybrid Systems, *Proceedings of the IMACS Symposium on Mathematical Modelling*, 1997, pp. 17-32.

[5] R.S. Sreenivas and B.H. Krogh, On Condition/Event Systems with Discrete State Realizations, *Discrete Event Dynamic Systems: Theory and Applications*, Vol. 1, 1991, pp. 209-236.

[6] K.E. Arzen, Using Multi-view Objects for Structuring Plant Databases, *Intelligent Systems Engineering*, 1993, pp. 183-200.

[7] S.E. Mattson, Simulation of Object-Oriented Continuous Time Models, *Proceedings of the IMACS Symposium on Mathematical Modelling*, 1994, pp.85-88.