Pergamon

Computers chem. Engng Vol. 22, Suppl., pp. S887–S890, 1998
© 1998 Elsevier Science Ltd. All rights reserved
Printed in Great Britain
0098-1354/98 $19.00 + 0.00

PII: S0098-1354(98)00173-2

Simulation support for recipe driven process operation

G. Mušič, D. Matko

Faculty of Electrical Engineering, University of Ljubljana, Slovenia

Abstract

Flexible batch plants are characterised by combination of discrete and continuous dynamics. Several processes taking place inside batch reactors are best described by continuous dynamics. The sequence of steps required to make a product, on the other hand, can be very well described by discrete event dynamics system representation. Such sequences are defined by batch recipes. A design methodology for such recipe driven process operation based on recipe models is proposed. Supervisory control framework is employed to combine procedures defined by recipes with co-ordination and resource-allocation functions. A simulation support for the proposed design approach is discussed and an example of a corresponding simulation tool is presented. An application example of a batch process cell controller design is presented to illustrate the described concepts. © 1998 Elsevier Science Ltd. All rights reserved.

Keywords: modelling, simulation, discrete event systems, Petri nets, process control, batch processing

INTRODUCTION

Flexible batch plants are a subject of increasing research interest in the last years. They feature several interesting properties from the system theory point of view. One of them is a combination of discrete and continuous dynamics that is inherent in every batch process plant. Several processes such as filling/discharging of reactors, heating and cooling, etc., are best described by continuous dynamics. The sequence of process operations that are required to make a product, on the other hand, can be very well described by a discrete event dynamics system representation.

Such a sequence is defined by a batch recipe. This includes both the procedure of making a batch as well as all corresponding raw material quantities and other parameters. The procedural part of the recipe can be modelled as a discrete transition system and the corresponding discrete event system analysis and design methods can be used in development of an appropriate control strategy.

When designing any kind of control strategy, the simulation can be used as a valuable supporting tool. Batch control systems with the recipe driven process operation are no exception to that. For example, simulation can be employed in the non-formal verification of designed recipes. If both the discrete and the continuous phenomena can be included in the simulation model, the performance of the control system can be estimated. This enables the optimisation of the batch cycle duration, raw materials consumption, etc. The estimated system performance can be used as an input parameter to the resource allocation and scheduling strategy design procedure.

The paper focuses on a combined discrete/continuous simulation support for designing recipe driven process operation. Petri nets and Sequential function charts (SFC) are used for modelling batch recipes. A simulation environment based on Matlab/Simulink is presented, which enables the simulation of the SFC recipes. Petri nets framework can be used to design resource allocation strategy, which enables sharing of equipment between different recipes. This can be transformed to SFC and the designed system performance is then simulated within the presented simulation environment. The described concepts are illustrated by a batch process cell application example.

MODELLING OF RECIPE DRIVEN PROCESS OPERATION

The ISA SP88 standard defines four levels for the types of recipes that are used in a batch process. When dealing with a specific batch process cell, the standard differentiates between the master and the control recipes. The master recipe includes information about required process cell equipment, raw materials with corresponding quantities, and the procedure of making a product. It is designed to be used on different lines within the process cell. The control recipe is created from the master recipe when a batch is scheduled for production. It includes additional information about the process units that are used in manufacturing a single batch of a specific product. In the presenting paper we focus on the procedural part of the recipes and how this is combined with the resource allocation strategy to derive a control recipe for a particular batch.

Sequential behaviour defined by recipes can be modelled within various discrete event modelling frameworks. Typically, such behaviour is modelled by some kind of state transition formalism such as finite state machines or Petri nets. The later seem particularly suitable for modelling of recipe based operation because a Petri net model, compared to a simple state transition diagram, contains additional structural information. This information enables certain properties of the model to be examined without the problem of state explosion (Valette, 1995).

Petri net (PN) is a directed, weighted, bipartite graph consisting of two kinds of nodes, called places and transitions (Murata, 1989). In graphical representation, places are drawn as circles, and transitions as bars or boxes. An arc connects either a place to a transition or a transition to a place. A state of a PN is defined by marking, which assigns to each place a nonnegative integer. The assigned numbers are interpreted as a number of tokens in a particular place. The state is changed by firing a transition.

When dealing with batch processes, places in a PN model can represent an operation of a single device (e.g., on/off valve), a particular phase of a recipe, or state of a particular equipment module (e.g., a reactor can be idle, waiting, occupied). Transitions can be linked to various process events, such as: prescribed level or temperature in the reactor vessel reached, reaction finished, etc.

The representation of the corresponding operation sequences at the lower, basic system operation level is often given in the form of a Sequential Function Chart (SFC) or Grafcet. This is accepted as an IEC standard (IEC 848 and IEC 1131-3) and defined as one of the standard languages for programming of programmable logic controllers (PLCs). SFC inherited many of its features from the theory of Petri nets. More precisely, safe interpreted Petri net can be defined so that its input-output behaviour is the same as the input-output behaviour of the SFC (David, 1995). Its strong relation to Petri net theory enables a SFC to be directly redrawn from a Petri net model and the classical properties of Petri nets, such as marking invariants, can be applied also to SFC.

In this way, the recipe model can be derived within the Petri net modelling framework and the obtained model can be analysed for desired properties. Once the model is proved to fulfil desired goals it can be easily transformed into a logic controller program.

The described relation of SFC to the Petri net theory and the incorporation into the IEC standard suggest the use of SFC as a standard representation form of the operational sequences at the lower, basic system operation level. There is, however, also a need for a common representation format of the sequential functions at the higher, supervisory control level. It has been shown in literature that SFC can be effectively used for that purpose as well (Arzen, 1994). Since a PN model can be relatively simply transformed to a SFC, a Petri net supervisory design framework (Yamalidou et al., 1996) can be employed in designing higher level control strategies, e.g. resource allocation. Supervisory control synthesis techniques assure correctness of the derived solutions with respect to the given specifications and final verification effort can be reduced significantly.

An example of two process operations competing for a shared resource is shown in Fig. 1. Petri net model of the resource booking is given and the transformation to several SFC modules is indicated.

The proposed methodology for the design of the recipe driven process operation is to derive Petri net models of the recipes first. Models are then examined for the desired properties. The resource allocation strategy is derived within a PN supervisory control framework and the resulting PN models are transformed into a SFC specification of logic controllers for the individual units. Finally, the SFC modules are simulated in combination with the continuous process models to verify behaviour and estimate the performance of the designed system.

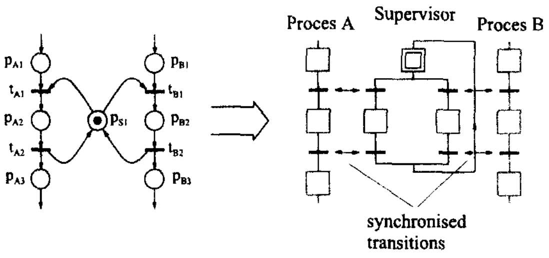

Figure 1: PN model and the corresponding SFC modules

SIMULATION SUPPORT

As mentioned before, the batch process cell is a mixed continuous/discrete system. Part of the discrete events, which enter the cell controller, is generated by the evolution of the continuous state. In the simulation of such combined system it is necessary to consider the behaviour of the continuous process in order to be able to generate state events - the discrete events resulting from the evolution of the continuous part of the overall system. The corresponding simulation algorithm has to cope with discontinuities, state event handling, event scheduling, state re-initialisation, etc. (Barton and Pantelides, 1993), (Cellier et al., 1993), (Wölhalf et al., 1996).

SFC tool for Matlab-Simulink

Our recent work concentrates on enhancements of the widely used continuous simulation tool Matlab-Simulink toward combined simulation capabilities. The emphasis of our work has been on development of a blockset that facilitates the design of the discrete part of the simulation model and its integration into the Simulink environment. A simulation mechanism for the discrete event model, which runs in parallel to the standard ODE solver, has also been implemented.

In our simulation approach the combined system is modelled as a hybrid system where the discrete and the continuous part of the system are separated and an interface is introduced in between (Antsaklis et al., 1993). The continuous process part is simulated by the existing Simulink continuous simulation block library. All continuous sub-processes are simulated there and the continuous controllers are included in the simulation scheme as well. The discrete control logic is in our solution described by a Sequential Function Chart, which enables use of Petri nets as a basic modelling framework, and at the same time serves as an intermediate step to industrial implementation of the derived solutions.

The SFC part of the simulation scheme is built of steps and transitions. Actions are assigned to steps and conditions are assigned to transitions. In our current implementation, actions are limited to change of the value of a single Boolean variable only, while we retained all action types defined by the standard. An action is defined inside SFC step by specifying a variable name followed by the action

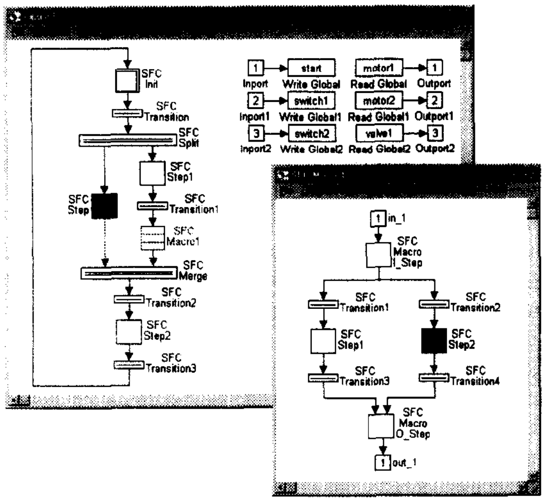

Figure 2: Macro-step and its expansion

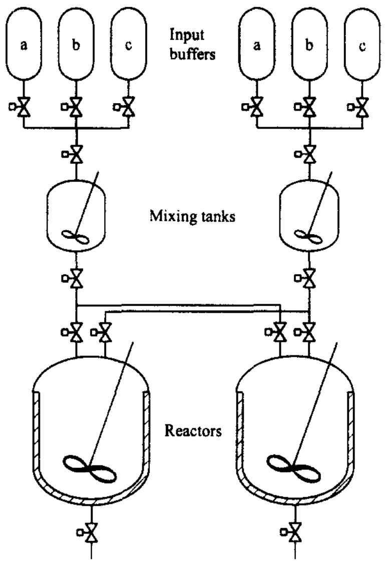

Figure 3: Batch process cell

type qualifier and an optional parameter. The values of action variables are then assigned to the discrete subsystem outputs.

Similarly, the communication between discrete subsystem inputs and the SFC transition blocks is also performed by assigned variables. Each input is assigned to a variable and expressions built up of these variables and standard Matlab operators are assigned to transitions. The result of such an expression is interpreted as a Boolean value that enables or disables the corresponding transition. The SFC part forms a separate subsystem block which is then linked to the rest of the scheme by its input and output signals as any other Simulink block.

Macro-steps, which contain a portion of the SFC defined elsewhere, can be used to make the simulation schemes of the complex sequential control strategies more transparent. The SFC that corresponds to a macro-step is defined in a separate window and is composed as an ordinary SFC with the following limitations: it must contain exactly one input and one output step and must not contain any additional I/O ports. An arbitrary number of nesting levels is supported, with the limitation that all variables used in steps and actions are global and care must be taken not to duplicate variable names. An example of the macro-step and its expansion is shown in Fig. 2.

The interface between continuous and discrete section consists of two parts. The first part performs the generation of events, which trigger the transition enabling conditions. State events are generated by comparison of the signals of interest to the specified threshold values. The result of each comparison is fed as input into a discrete subsystem. The second part of the interface maintains the processing of the results of the discrete actions. The discrete subsystem outputs are appropriately transformed and can be used in the continuous part of the simulation scheme as switching signals or step shaped inputs.

APPLICATION EXAMPLE

A modelling and simulation case study of a batch process cell is presented in this section to illustrate the discussed concepts. The process cell is shown schematically in Fig. 3. It consists of several input buffers, two mixing tanks and two reactor vessels. Each reactor has two inlets for two incoming substances and the output is produced by the reaction of the two chemicals at a specified temperature. The filling of the reactor is controlled by the two on/off valves, and the discharging is controlled by the on/off drain valve. The maximum and minimum levels are sensored by two level switches. The temperature of the reaction is controlled by a flow of hot or cold water through the water jacket, which surrounds the reactor vessel. The water flow is controlled by adjustment of the corresponding proportional valves and the temperature of the reactor contents is measured by continuous temperature sensor.

The preparation of the input substances takes place in two mixing tanks that are supplied by the raw material from three supply tanks each. The substance is composed from one of the two basic components (component 'a' or component 'b') diluted to required concentration by component 'c'. The filling and discharging of the mixer is controlled by the set of on/off valves. The level in the mixing tank is measured by the continuous level sensor, required quantities of each input depend on the recipe.

Because of the two reactors, two process lines can be identified in the cell. In other words, two batches can be processed in the cell simultaneously. However, both batch recipes use both mixers to prepare the input substances. The two batches therefore compete for the two mixing tanks. We will focus on the mixing part of the cell and show how the conflicts can be avoided by designing an appropriate supervisory mechanism within the Petri net framework.

The simplified procedural part of the two recipes is depicted in Fig. 4. The corresponding procedures are identical, with the control interpretation of places and transitions given in Tab. 1. The shared resources are the two mixing tanks. If we assume that a new batch may be scheduled before the current batch is finished, the two reactors are shared as well. Beside that, the combination of two

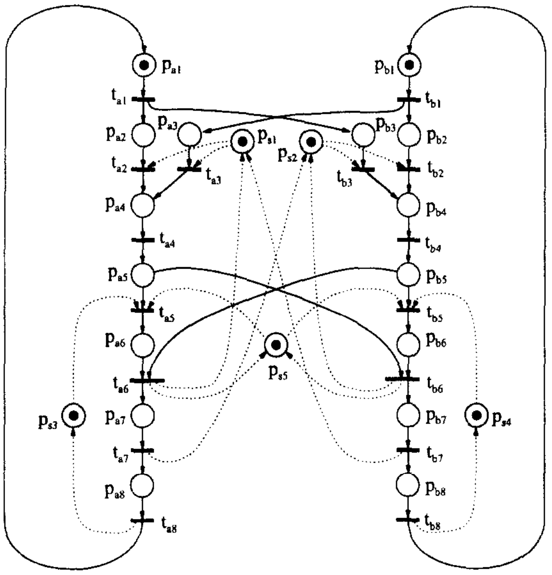

Figure 4: Concurrent run of two batch recipes

Table 1: Control interpretation of places and transitions

Places	
p_{a1}, p_{b1}	reactor ready
p_{a2}, p_{b2}	mixing request A
p_{a3}, p_{b3}	mixing request B
p_{a4}, p_{b4}	mixing (a, b, c)
p_{a5}, p_{b5}	substance ready
p_{a6}, p_{b6}	filling A
p_{a7}, p_{b7}	filling B
p_{a8}, p_{b8}	heating, reaction, cooling, discharging

Transitions	
t_{a1}, t_{b1}	new batch scheduled
t_{a2}, t_{b2}	start preparing substance A
t_{a3}, t_{b3}	start preparing substance B
t_{a4}, t_{b4}	required level reached
t_{a5}, t_{b5}	start filling the reactor
t_{a6}, t_{b6}	mixing tank A empty
t_{a7}, t_{b7}	mixing tank B empty
t_{a8}, t_{b8}	reactor empty

recipes as shown in Fig. 4 leads to a deadlock when transitions t_{a5} and t_{b5} are fired simultaneously. An additional supervisory mechanism is required to prevent such a deadlock. The dotted arcs denote connections of recipe transitions to the supervisory places. These are derived from the PN model of the recipes by the method of place invariants (Yamalidou et al., 1996). Places p_{s1} and p_{s2} perform the booking of the two mixers and places p_{s3} and p_{s4} perform the booking of the reactors. Place p_{s5} prevents a deadlock in the system.

The derived model is transformed to several SFC modules, similarly as shown in Fig. 1. The SFC modules actually represent the logic controller program and the supervisory algorithm. The corresponding simulation model is built by the developed Simulink SFC blockset and linked to the continuous model of the processes describing level and temperature inside reactor and mixing vessels. The behaviour of the controlled system is then simulated in the proposed simulation environment. The graphic animation capabilities of Matlab are used to present the simulated model behaviour.

CONCLUSIONS

The presented extension of a continuous simulation tool enables verification of the desired behaviour and evaluation of the system performance in presence of various disturbances and changes in the system. The simulation environment proved itself valuable in testing various sequential control strategies and can be used as a general purpose sequential control logic program testing environment. The tool is particularly effective when the sequential control is applied to continuous processes, such as in the case of batch processing. The application of the presented tool is not limited to off-line simulation only. Matlab open architecture enables the exchange of data with other program packages. An interface to the FactoryLink SCADA package, which will enable the communication between Matlab-Simulink and a real-time process database, is under development. In proposed control system structure, the SCADA package will cope with the process visualisation and communication to the lower level process control equipment while Matlab-Simulink will be used to perform on-line advanced supervisory functions.

REFERENCES

Antsaklis, P. J., Stiver, J. A. and Lemmon M., 1993, Hybrid System Modeling and Autonomous Control Systems. In: R.L. Grossman, A. Nerode, A.P. Ravn, H. Rischel, eds., *Hybrid Systems*, Lect. N. in Comp. Sci. **736**, Springer, Berlin, pp. 366-392.

Arzen, K. E., 1994, Grafcet for Intelligent Supervisory Control Applications. *Automatica*, **30**, pp. 1513-1525.

Barton, P. I. and Pantelides, C. C., 1994, Modelling of combined discrete/continuous processes. *AIChE*, **40**, pp. 996-979.

Cellier, F. E., Elmquist, H., Otter, M. and Taylor, J. H., 1993, Guidelines for Modelling and Simulation of Hybrid Systems. In: *Proc. IFAC 12th Triennial World Congress*, Vol. 8, Sydney, pp. 391-397.

David, R., 1995, Grafcet: A Powerful Tool for Specification of Logic Controllers, *IEEE Trans. on Control Systems Technology*, **3**, pp. 253-268.

Murata, T., 1989, Petri Nets: Properties, Analysis and Applications. *Proc. IEEE*, **77**, pp. 541-580.

Valette, R., 1995, Petri nets for control and monitoring: specification, verification, implementation. *Proc. of Workshop on Analysis and and Design of Event-Driven Operations in Process Systems (ADEDOPS)*, Imperial College, London, 1995.

Wölhalf, K., Fritz, M., Schultz, C. and Engell, S., 1996, BaSiP - Batch Process Simulation with Dynamically Reconfigured Process Dynamics. *Computers Chem. Engng.*, **20**, Suppl., S1281-S1286.

Yamalidou, K., Moody, J., Lemmon, M. and Antsaklis, P., 1996, Feedback Control of Petri Nets Based on Place Invariants. *Automatica*, *32*, pp. 15-28.