A MODULAR BATCH LABORATORY PROCESS

Rasmus Olsson and Karl-Erik Årzén

Department of Automatic Control
Lund Institute of Technology
Box 118, SE-221 00 Lund, Sweden

Abstract: In this paper a new batch laboratory process is described. The process is inexpensive, modular, and portable. Several processes can be connected together to form a multi-purpose batch pilot plant. The use of the process in process control education is described.

Keywords: Batch Control, Laboratory Processes, Control Education

1. INTRODUCTION

A good control engineering course should include hands-on experiments. A number of laboratory control processes are available commercially, e.g., inverted pendulums, helicopter processes, level-controlled water tanks, etc. These are commonly used in laboratory exercises in different basic and advanced control courses. However, very few processes exist that illustrate the typical problems associated with batch control.

In this paper a new laboratory batch tank process is presented. The process can be used stand-alone for illustrating the control problems associated with single-unit batch control. It is also possible to connect several processes together to form a multi-purpose batch cell. Used in this way the process can be used in teaching laboratories on recipe-based batch control and scheduling of batch processes.

The process consists of a water tank equipped with inlet and outlet pumps, a heating device, a cooling device, and an agitator. The sensors consist of a level sensor and a temperature sensor. The relatively large amount of actuators means that the number of discrete operations that can be performed is quite large (start/stop inlet pump, start/stop heating, etc). The actuators can also be controlled with analog signals. This means that the tank also can be used as a continuous jacketed tank reactor. The possibility to both cool and heat can be used to emulate the temperature behavior of different types of chemical reactions, e.g., the cooler can be used to simulate an endothermic reaction in the reactor. In this way the simulation of the reaction rate can be made temperature dependent, non-linear, or constant. The process can also be used for multi-variable control experiments, e.g., simultaneous control of level and temperature.

The process is made from standard components and therefore inexpensive. The process is also small, so small that it fits easily on the desk beside a standard PC used for control. The connection between the process and the PC is an ordinary RS-232 serial line. This also means that it is straightforward to control the process from an ordinary laptop PC at lectures and presentations.

[1] This work has been supported by the Center for Chemical Process Design and Control, CPDC, and the Swedish Foundation for Strategic Research, SSF. The development of JGrafchart is funded by the EC/GROWTH project CHEM aimed at developing advanced decision support systems for different operator-assisted tasks, e.g., process monitoring, data and event analysis, fault detection and isolation, and operations support, within the chemical process industries. The CHEM project is funded by the European Community under the Competitive and Sustainable Growth Programme (1998-2002) under contract G1RD-CT-2001-00466.

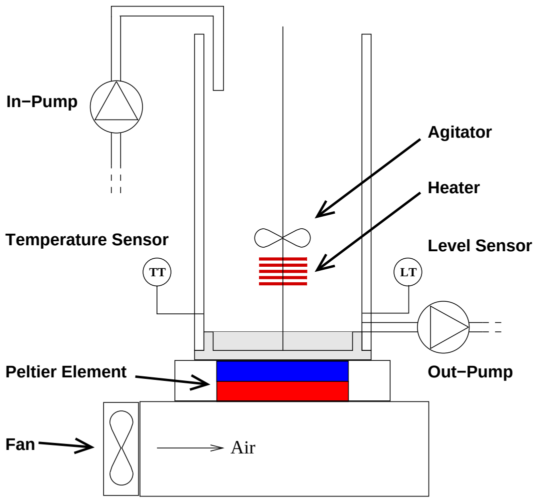


Fig. 1. Outline of the laboratory batch process.

1.1 *Outline of the Paper*

The process is described in detail in Section 2. Currently the process is used in the basic course in process control at the Department of Automatic Control, Lund Institute of Technology. Section 3 describes the laboratory exercise. A dynamic simulator for the batch tank process is also described. In the laboratory exercises the process control system is implemented in JGrafchart, a Java-based environment for graphical programming that combines the functionality of Grafcet/SFC from IEC 61131-3, with Statecharts and procedural and object-oriented programming concepts. A brief overview of JGrafchart is given in Section 4. By combining three tanks in series a multi-purpose batch cell can be emulated. This is described in Section 5.

2. PROCESS DESCRIPTION

The batch tank consists of a plexi-glass tube with a brass bottom. The volume of the reactor is approximately 0.5 liter. Connected to the reactor is an agitator, an in-pump, an out-pump, a level sensor, a temperature sensor, a heater, and a cooler, see Fig. 1. The cooler consists of a 40 W Peltier element cooled by a fan.

The outputs from the level and temperature sensors are in the range $0-10$ V, which corresponds to Empty-Full tank and $0-100^o$ C respectively. The resolution OF THE SENSORS can be up to 10 bits.

The pumps, heater, and cooler can either be controlled with digital, i.e. On/Off, signals or analog signals. The analog signal is transformed using an integrated microcontroller for pulse width modulation (PWM). The built-in microcontroller is an 8-bit ATmega8 from Atmel. The agitator only has a digital control signal, but to overcome the problem with friction the microcontroller is programmed to give a higher control signal right when the agitator is turned on. The heater is a 24 V, 150 W heating element. There are three light-emitting diodes (LED) on the front of the process: one red for showing if the heater is on, one blue for showing if the cooler is on, and one red/green LED, which is red if there is some error and green if the power is on and the process is OK.

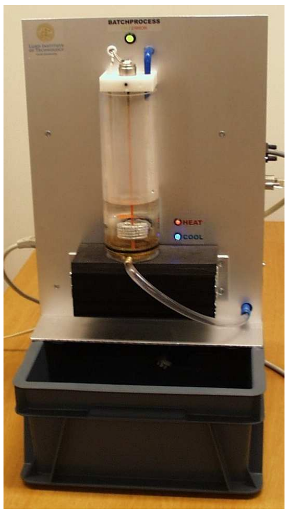


Fig. 2. The real process.

The microcontroller is used for safety interlocks, e.g., it is not possible to turn the heater on if the tank is empty, the in-pump will stop once the tank is full, and there is a protection against over heating. The microcontroller enables communication over an RS-232 serial line. This makes it possible to control the process with a laptop, e.g., at presentations and lectures, without using an IO card.

The water container below the process is an off the shelf plastic tank with the size: 15 cm high, 30 cm wide, and 40 cm deep. The container also becomes a protection for the front part of the process during transportation. The real batch process can be seen Fig 2.

The cost of building one process, without the development cost, is approximately US$1200, of which half is material and half is work cost.

3. A BATCH PROCESS LABORATORY EXERCISE

The process is used in a laboratory exercise in the course “Automatic Process Control”, a fourth-year course for chemical engineering students at the Lund Institute of Technology. The laboratory exercise introduces the students to discrete-event control of a batch reactor and it also requires the students to implement a discrete PID-controller.

Both the sequential control and the PID-controller are implemented in JGrafchart. The sequential control consists of the following steps:

I) Fill the reactor.
II) Heat the reactor, during simulation of an endothermic reaction, to a specified temperature using PID-control.
III) Empty the reactor.
IV) Clean the reactor.

The sequence should then be restarted and a new batch made. A solution in JGrafchart can look like in Fig. 4.

In the exercise the cooler is used to simulate an endothermic reaction in the reactor. This way the simulation of the rate of the reaction can be made temperature dependent, non-linear, or constant.

3.1 *Process Simulator*

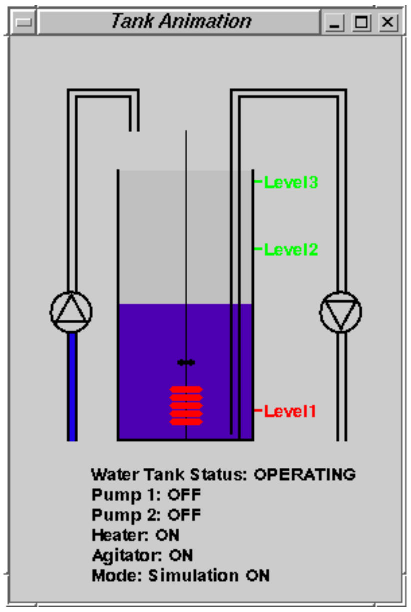
During the design of the hardware the temperature dynamics of the process was estimated to be a first order system with time constant $T \approx 1000$s (if the temperature step is not too large). The laboratory exercise would take a very long time if all the experiments should be made on the real process and therefore a simulation model has been developed. The simulation model can be used for development of controller schemes and parameter adjustments. The speed of the simulation can be changed.

To be able to plot signals in real time for the real process it was decided to construct a simulator server program that offers the following services:

a) A simulated batch process with the same interface as the real batch process.
b) Possibilities to plot temperature measurements, control and reference signals.
c) An animation of the batch process.
d) A socket interface so that clients can acquire all of the above services.

It was decided to implement the simulator server in Java. Two communication threads were implemented. One that sends measurement signals from the virtual tank to clients connected to the server, and one that receives commands from clients.

The simulated batch process was implemented as a subclass of an existing process simulation class package. The class package provides methods to simulate systems that are described by differential or difference equations in real time. The coefficients for the differential equations were identified from the real process. The class also provides methods to animate the process with the public Graphics2D-class package. A screen-shot from the simulation is shown in Fig. 3.



```
Water Tank Status: OPERATING
Pump 1: OFF
Pump 2: OFF
Heater: ON
Agitator: ON
Mode: Simulation ON
```

Fig. 3. Animation of the tank used both for the simulated and the real process.

A client, which connects to the server's socket can switch the simulation on and off. If the simulation is on the virtual process will react to the client's command signals and visualize the effect in the animation window. Moreover the signals will be plotted. If the simulation is turned off only the animation and plotting capabilities of the server are used to plot the signals from the real process.

4. JGRAFCHART

JGrafchart is the name of a Java-based version of Grafchart (Årzén, 2002), an object-oriented extension to Grafcet/Sequential Function Charts. JGrafchart consists of a graphical editor and a run-time system. In the graphical editor the user creates Grafchart sequential function charts by copying language elements from a palette using drag-and-drop. The language elements are placed on a workspace and connected together graphically. After compilation the function charts are executed by the runtime system within the JGrafchart machine. No native code generation takes place. A separate execution thread is used for each top-level workspace. The execution is periodic. In every cycle the inputs are read, a scan of the function charts is performed, and the outputs are written. JGrafchart is shown in Fig. 4. The drag-and-drop palette is shown to the left. JGrafchart is implemented in Java 1.4 and Swing. The graphical object editor class package JGo from Northwoods Software Corporation is used in the implementation of the graphical editor. The public domain parser generator JavaCC is used to generate parsing code for all textual expressions. JavaCC is also used to build abstract syntax trees. Storage to file is provided in the form of XML. The

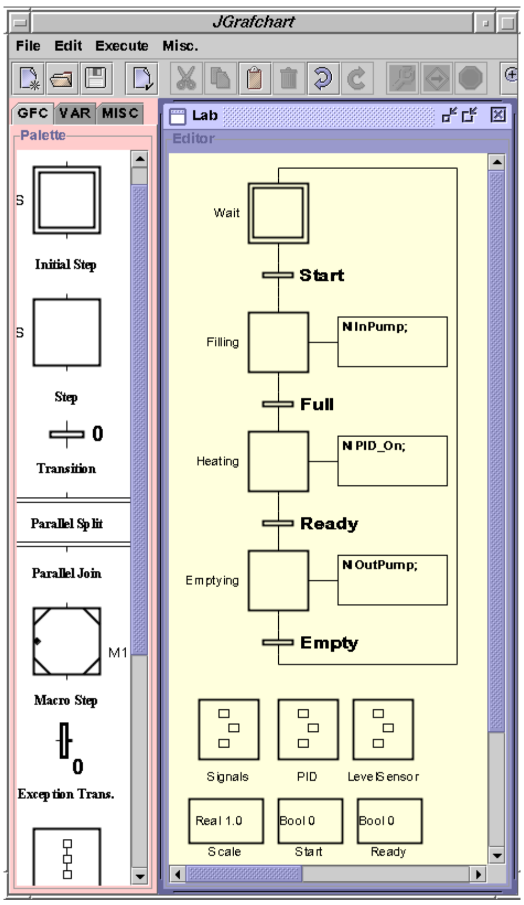


Fig. 4. Control sequence in JGrafchart

contents of each document can be stored as XML and later reloaded. It is also possible to store all top-level workspaces as a single XML file.

The main language elements are steps, transitions, macro steps, procedures, and procedure steps. A step represents a state where actions are performed. Four types of actions are supported: stored actions are executed when the step becomes active, exit actions are executed immediately before the step becomes deactivated, periodic actions are executed periodically while the step is active, and, finally, normal actions which are used to associate the truth value of a boolean variable with the activation status of the corresponding step. Transitions contains a boolean condition and/or event expression that decides when the transition should be fired. The grammar for transition expressions and step actions is defined formally using JavaCC.

JGrafchart supports four different data types: booleans, integers, reals, and strings. Using special workspace objects it is possible to create complex variables, e.g., structs. Workspace objects can also be used to encapsulate procedures, e.g., behavior. In this way it is possible to mimic simple object structures. JGrafchart can interact with the external environment in four different ways: using digital IO, using analog IO, using sockets, and using XML based communication. Using digital inputs and outputs JGrafchart can be used directly as a logical controller. Using analog input and output blocks JGrafchart can communicate with A-D and D-A converters. This functionality only works when executing on the Linux PCs in our laboratory. Each workspace can act as a client in a TCP socket connection connecting to some server, e.g., a simulator. Using socket input and output blocks it is possible to read and write variable values using a simple text-based protocol where each test message has the following structure: `variable identifier '|' value`. Finally, using XML input and output blocks, JGrafchart can communicate with xmlBlaster, an XML-based message-oriented middle-ware layered on top of XML-RPC or CORBA. The xmlBlaster server supports communication using the publish/subscribe method and using remote procedure calls.

The graphical editor provides a number of features, e.g., graphical selection of objects, move, delete, cut-copy-paste, undo-redo, change object size, group-ungroup, move-to-front, send-to-back. It is possible to change the grid size and color of a workspace. In addition to the Grafchart language elements the editor supports graphical objects, e.g., texts, rectangles, ellipses, icons, buttons, etc. Step actions can be associated with a button and executed when the button is pressed. It is also possible to dynamically change the attributes (position, size, color, and so on) of the graphical objects using step actions. In this way it is possible to implement GUIs in the form of animated process diagrams.

5. A LABORATORY BATCH CELL

The mixed discrete/continuous nature of batch production systems makes them a challenging control problem. In multi-purpose batch cells it is possible to produce multiple products at the same time using a finite set of equipment units. The units are organized in a network structure with a high degree of connectivity. The sequential information about the ordering of the operations, and the equipment unit types required for the different operations, is captured in a recipe. In order to execute an operation in an equipment unit, the batch control system must allocate the resource that the equipment unit constitutes. A control system for recipe-based batch production must, hence, include substantially more functions than what is needed in a control system for continuous production. An example of a batch pilot plant is the Procel plant at UPC in Barcelona, Spain. Procel has been used in several ways in the research of batch control, see e.g. (Cantón *et al.*, 1999; Ruiz *et al.*, 2001; Olsson, 2002).

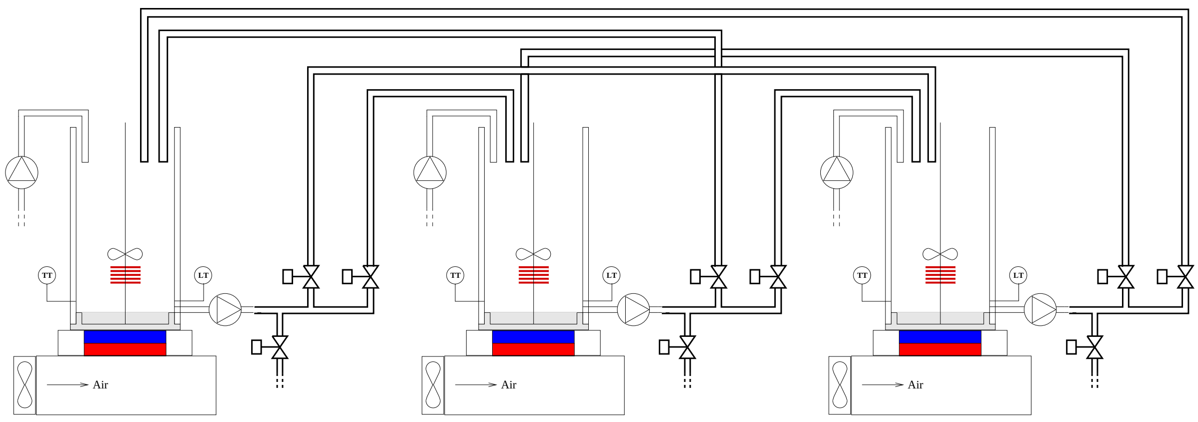


Fig. 5. Outline of the batch pilot plant consisting of three processes.

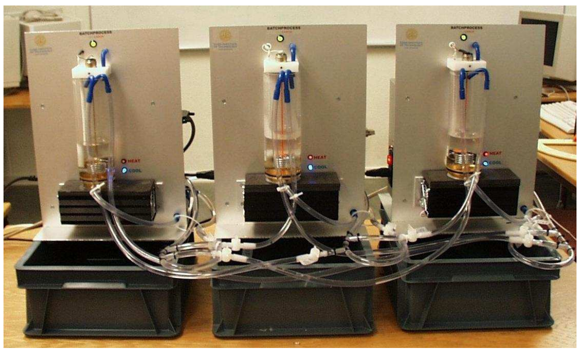


Fig. 6. Tank processes connected together to a small batch pilot plant.

Batch cells are by nature quite large (Procel is 4 m by 2 m by 1 m) and expensive, and not so well suited for teaching laboratory purposes. However, by connecting the developed single-tank processes together a small pilot batch cell plant can be created. As a test three processes have been connected together using plastic pipes and manual valves, see Fig. 5 and Fig. 6. The manual valves can easily be replaced with automatic magnetic valves if desired. The plant is highly flexible and connected in such a way that each of the tanks can transfer its content to any of the other tanks or empty it out to the water container below. The plant has the same functionality as the Procel pilot plant.

The control system is implemented in JGrafchart according to the S88 batch control standard (ISA, 1995; IEC, 1997). Each tank is modeled as an equipment unit in the control system. The equipment unit contains equipment operations or phases for performing the different control actions on the physical batch tank unit. The recipes are represented as JGrafchart procedures consisting of sequences of recipe operations or recipe phases. The recipe operations or phases call the corresponding equipment operations or phases using method calls. In JGrafchart it is also possible to include an animated process diagram that can be used as operator interface. The JGrafchart

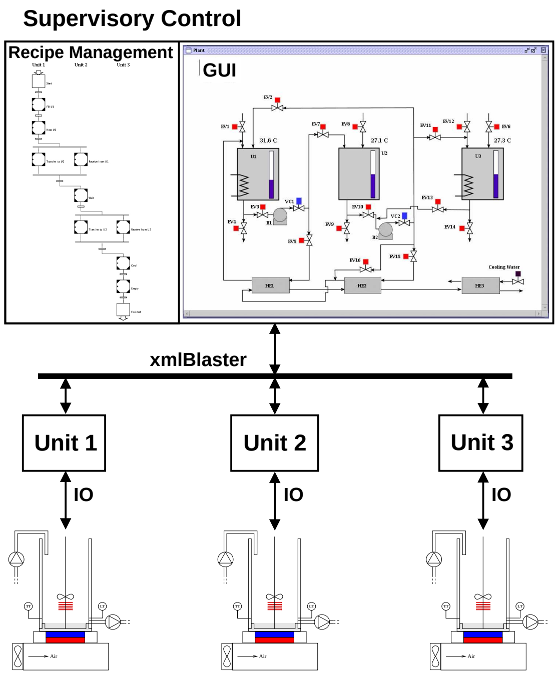
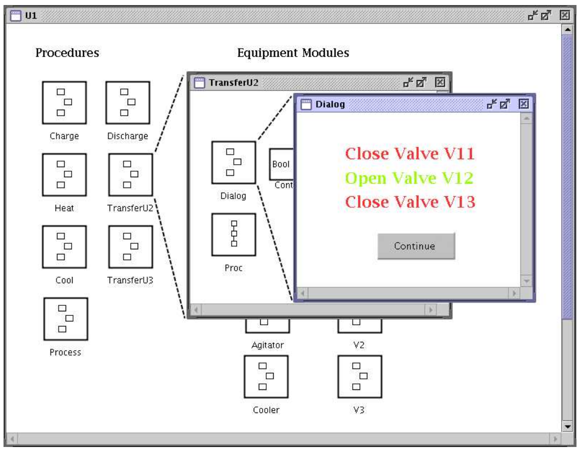


Fig. 7. Distributed control system according to the batch control standard S88.

solution can either be centralized executing on a single PC or distributed executing as multiple JGrafchart applications either within the same PC or on different PCs, e.g., one PC for each equipment unit and one supervisory PC for the recipe handling and the operator interface, see Fig. 7. The distributed solution relies on the possibility in JGrafchart to communicate using XML-messages.

To include the manual valves in the control system, dialog windows requesting the operator to manually open and close the valves have been added. Instead of opening the valve automatically it brings up the dialog and waits for the operator to confirm that the valves are in the correct position. A dialog window can be seen in Fig. 8. How JGrafchart can be used for operator support is described in (Årzén *et al.*, 2002). If automatic magnetic valves are added to the plant they can



Close Valve V11
Open Valve V12
Close Valve V13

Fig. 8. Dialog window in JGrafchart for manual control of valves in a transfer operation

either be considered to be a part of the existing units or they can form a separate valve-battery unit. If the valves are grouped as a new unit the control system for the single tank process can be re-used without change. New operations need to be added for the transfer from one process to another but to a large content this could be part of the valve-battery unit. This way the plant could easily be extended to more units over time, only the control of the valve battery needs to be changed.

A recipe for the manufacture of a certain product in the plant may for example consist of the following sequential operations:

- Fill Unit 1.
- Heat Unit 1.
- Transfer to Unit 2.
- Wait.
- Transfer to Unit 3.
- Cool Unit 3.
- Empty Unit 3.

The recipe implemented as a procedure in JGrafchart is shown in Fig.9.

6. CONCLUSIONS

An inexpensive, portable, and flexible laboratory batch process has been developed. The process is used within the chemical engineering education at Lund Institute of Technology. The process can be used for teaching sequential, PID, multi-variate, and recipe-based control. It has also been used in the development of an exception handling strategy for batch control (Olsson, 2002).

A major advantage of the modularity and size of the process is that it can easily be disconnected and stored or transported. Once disconnected each process can be fitted into a bag and easily carried by one person.

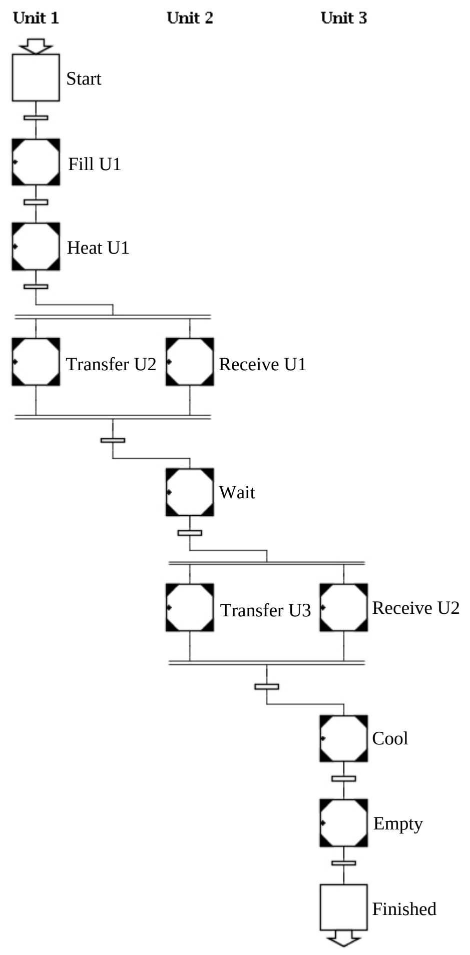


Fig. 9. A recipe implemented in JGrafchart

7. REFERENCES

Cantón, J., D. Ruiz, C. Benqlilou, J.M. Nougués and L. Puigjaner (1999). Integrated information system for monitoring, scheduling and control applied to batch chemical processes. In: *Proc. of the 7th IEEE Int. Conf. on Emerging Technologies and Factory Automation.*

IEC (1997). IEC 61512-1: Batch control, Part 1, Models and termonology. Technical report. International Electrotechnical Commission.

ISA (1995). ISA S88.01 batch control. Instrument Society of America.

Olsson, Rasmus (2002). Exception Handling in Recipe-Based Batch Control. Licentiate thesis ISRN LUTFD2/TFRT–3230–SE.

Ruiz, D., J. Cantón, J.M. Nougués, A. Espuña and L. Puigjaner (2001). On-line fault diagnosis system support for reactive scheduling in multipurpose batch chemical plants. *Computers and Chemical Engineering* **25**, 829–837.

Årzén, Karl-Erik (2002). JGrafchart. Home page: http://www.control.lth.se/∼grafchart/.

Årzén, K.-E., R. Olsson and J. Åkesson (2002). Grafchart for Procedural Operator Support Tasks. In: *Proceedings of the 15th IFAC World Congress, Barcelona, Spain.*