

Open Architecture Modeling and Simulation in Process Hazard Assessment

Mordechai Shacham

Dept. of Chemical Engineering, Ben Gurion University of the Negev, Beer-Sheva 84105, Israel; e-mail: shacham@bgumail.bgu.ac.il

Neima Brauner

School of Engineering, Tel-Aviv University, Tel-Aviv 699 78, Israel;
e-mail: brauner@eng.tau.ac.il

Michael B. Cutlip

Dept. of Chemical Engineering, University of Connecticut, Storrs, CT 06269, USA
e-mail: mcutlip@uconnvm.uconn.edu

Abstract

Dynamic modeling and simulation of a chemical process in emergency conditions is considered. Such modeling and simulation often requires rapid implementation of model changes by the process engineer followed by simulation runs. A simulator with an open architecture structure is presented where the model components (equations) are stored in an object-oriented form in a database, enabling rapid and easy modification of the model. A general-purpose numerical solver is used for solving the model and plotting the pertinent results. An example is presented, where the use of the proposed simulator enables a rapid identification of the right strategy to prevent development of runaway temperature conditions in a reactor.

Keywords

Process hazard analysis, dynamic simulation, open architecture, runaway reaction

1. Introduction

Dynamic modeling and simulation of chemical processes is being extensively used for quantitative hazard assessment (Srinivasan and Venkatasubramanian, 1999, Dhurjati, 1987) and in operator training for emergency situations. Process hazard assessment (PHA) involves several steps, which include hazard identification, assessment of the likelihood of failure and estimation of the potential for damage and/or injury associated with specific incidents. Dynamic simulation is a very important component of PHA, since it can quantitatively predict the consequences of critical component failures. It is essential for operator training, as in reality process faults rarely happen and intentional creation of emergency situations for training purposes is usually considered unacceptable.

The process models that are being used today for PHA have a rigid structure where the process model cannot be changed

and faults and disturbances can be introduced only with respect to a limited number of variables (Dhurjati, 1987). The process engineer typically does not have access to the process model of the simulator. A change of the model requires involvement of a computer programmer since the process model equations are coded in a programming language. Furthermore, any change of the model requires an additional cycle of testing and debugging the simulator.

Simulation is mostly needed in emergency situations, since taking the correct course of actions (based on simulation results) can prevent culmination of an unpredicted event into a major disaster. Fogler (1999, pp. 542-547) analyzed, for example, the "Nitroaniline reactor rupture" incident that happened in Saugnet, IL in 1969. He has shown that a simulation study of the reactor, which was operated at that instance under abnormal conditions, could have predicted the runaway of the temperature at a later stage of the reaction. Using this information, the operators could have initiated the

required actions to reduce the temperature of the reactor, thus preventing its ultimate rupture.

The rigid structure of a simulator severely limits its applicability to emergency situations. Emergency usually arises due an unpredicted event that, most probably, has not been programmed into the simulator. Thus, in emergency situations, intervention of the process engineer in the mathematical model level may be required. In such situations, however, there is usually no time and experienced personnel may not be available to carry out major program changes.

The current trend in process simulation is toward life cycle modeling and the use of open system architectures for process simulation (Marquardt et al, 1999, Zaarur and Shacham, 1999). The potential benefits related to process safety offered by using an open architecture simulation are that the mathematical model can be easily and inexpensively modified to reflect operational or structural changes carried out during a life cycle of the process. The process engineer can later introduce faults and failures into the model that were not anticipated at the design stage while avoiding major program changes.

In this paper, the structure of an open architecture process simulator intended for safety applications is presented and its use is demonstrated for the case of a temperature runaway in a polymerization reactor.

2. The structure of an open architecture process simulator

Zaarur and Shacham (1999) describe the structure of an open architecture process simulator. The main difference between the traditional and the open architecture structure is that in the open architecture structure the simulator may consist of several independent components (layers), where the communication between the various components is restricted to well defined channels. Different software packages can be used for the different components and a component can be revised or updated without affecting the other components. A short description of the tasks of the various components, their structure and connections with the other components follows.

User Interface

The communication between the users and the simulator is done solely through the user interface (UI). To make the simulator useful for non-specialized users, the UI options must be self-evident, easy to learn for a first time user and easy to relearn for occasional users.

On problem input, the UI must enable the user to load and modify an existing process model, change input data, such as operational conditions and parameters of the unit operations, change/add physical and thermodynamic properties, or build in a new process model. On output, the UI should provide the

user with the flexibility to present the results related to certain variables as requested, in various tabular and graphical forms. These results should be available for display, printout and copying or transferring to other programs.

On problem input, the mathematical model and all the respective data are transferred to the data base management system, which takes care of the storage and retrieval of the model and the data. On output, previous simulation results can be retrieved from the database.

Mathematical model library

The models of the unit operations as well as physical and thermodynamic property correlations are stored in a database. The sub models that are used in constructing the model are equations of the form: output variable = g (input variables, output variable and constants) where g is a function. The equations are ordered and aggregated according to principles of model building. First added to the model are all the basic balance equations. Next, the input variables of these equations are specified. Some of the variables should be expressed as constitutive equations (e.g. heat and mass transfer rates, reaction rate) others as thermodynamic property correlations or constants. The addition of new equations is continued as long as there are still input variables that have been defined as output variables.

Note that the exact format of the equation depends on the type of the equation to be solved (differential, algebraic etc.) and on the numerical solver which is used. The format of the equations must be such that they are interpretable by the numerical solver. On the other hand, the format and syntax of the equations must be as close as possible to the common mathematical format so that the user does not have to learn complex syntax rules for building a model. The equations format will be discussed in more detail in the next section.

Object-oriented techniques allow the representation of detailed knowledge about the various equations. Information strings are attached to each of the equations in the database, which include: 1. A unique index to identify the equation, 2. A description of the output variable, including units, 3. An initial value or an initial estimate (as required) 4. A definition of the equation in the form that was shown earlier, and 5. A brief explanation of what the equation represents.

Numerical solver

Mathematical models of chemical processes can be categorized as systems of nonlinear algebraic equations, systems of ordinary or partial differential equations and differential algebraic systems. The selected numerical solver should contain programs for solving large systems of those types of equations. The numerical solver receives the model

equations, the initial, final or boundary values from the model library. It checks the equations for consistency and attempts to solve the model when it is well defined. The solution is transferred to the data base management system, which enables the user to define the variables to be displayed, printed, plotted or transferred.

3. Implementation of the open architecture simulator for process hazard assessment

The proposed implementation of the open architecture simulator for process hazard assessment includes the use of a spreadsheet program (Excel¹) as the user interface on input and the database management system, and Polymath 5.0² as a numerical solver and the user interface for output.

The advantage of Excel is that practically all engineers use a spreadsheet program, thus there is no need to learn to use it. It has the database management options that are needed for constructing process models according to the principles that were outlined in the previous section. However, for solving the model equations using Excel, the variables' names must be converted to cell addresses. Such a conversion requires writing a special compiler for this purpose. Furthermore, Excel has only a limited number of solution options. Therefore, as a numerical solver it is inappropriate.

There are several numerical software packages that can be used as numerical solvers. Shacham and Cutlip (1999) have recently compared six such packages for their appropriateness in chemical engineering education and found the Polymath package superior to the others in terms of user friendliness and the amount of technical effort involved in the solution. At the time when that survey was carried out, Polymath was limited to small scale educational type problems because of the limit on the number of equations it could handle. In the latest version of Polymath, the limit on the number of equations was removed; thus it can be used for solving industrial scale problems. This package has several robust algorithms for solving systems of nonlinear algebraic equations, several algorithms for solving stiff and non-stiff systems of ordinary differential equations. It can solve differential-algebraic systems using the controlled integration method (see Shacham et al, 1998) and partial differential equations using the "method of lines" (Cutlip and Shacham, 1999). It has many options for presenting, analyzing and regression of the simulation results. In dynamic simulation initial, minimal, maximal and final values of all the variables are presented, allowing easy identification of variables which

are out of their normal operating range. Several of the variables can be plotted versus any other variable, allowing easy discrimination between unstable or cyclic operation of the process.

The format for the equation input in Polymath follows the general format defined in the previous section and the usual engineering notation can be used for variable names. To allow the program to identify the type of the equation, special syntax is used in the left-hand side of the equations. An implicit algebraic equation $g(y, \mathbf{x}, \text{const})=0$, where y is the output variable of a particular equation and \mathbf{x} is a vector of input variables should be written: $f(y) = g(\mathbf{x}, \text{const})$. An ordinary differential equation should be inputted as: $df(y) = g(\mathbf{x}, \text{const})$. An explicit algebraic equation should be written in the form: $y = g(\mathbf{x}, \text{const})$. Using the "method of lines" for solving partial differential equations or the "controlled integration method" for solving differential algebraic systems enables bringing the equations into one of the forms described above.

The structure of the proposed simulator for process hazard assessment is schematically described in Figure 1. Excel is used as user interface for data input and as data base management system for creating and storing the process models. It is also used as archive of historical data, so that results of current simulations or process operation data can be compared to old data. Polymath is used as the numerical solver and as a user interface for output of results. Transferring the model from Excel to Polymath requires storage of the model column as an ASCII file in Excel and then reading the same file from Polymath. Tabular or graphical results can be transferred from Polymath to the data archive in Excel by simple "copy and paste".

It should be noted that Excel and Polymath are used only for demonstration of the proposed principle and different combination of programs can be used for constructing the simulator, according to the developer's preference. In the next section the use of the proposed simulator structure for simulation of a practical problem is demonstrated.

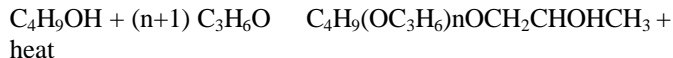
4. Propylene oxide polymerization reactor – an example

Propylene oxide polymerization is a highly exothermic process, which is carried out at high pressures. Nearly isothermal operation is required in order to prevent runaway conditions and the buildup of a pressure which is higher than the reactor's design pressure. Safety problems associated with the operation of such a reactor, are described in Kneale and Foster (1968). Mathematical modeling and simulation of the reactor, which includes a burst disk for pressure relief in case of excessive pressure buildup was carried out by Ingham et al (1994).

¹ Excel is a trademark of Microsoft Corporation (<http://www.microsoft.com>)

² POLYMATH is copyrighted by M. Shacham, M.B. Cutlip and M. Elly (<http://www.polymath-software.com>)

Ingham et al (1994) considered the manufacture of a polyol lubricant by step-wise condensation of propylene oxide with butanol:



A schematic description of the reactor for carrying out this reaction is shown in Figure 2. The catalyzed alcohol is initially charged into the reactor, up to the “initial level”. The oxide is fed into the reactor at a constant rate until the batch is ready and the reactor is full. Excess heat of the reaction is removed via an external heat removal system. Economical considerations dictate that the reaction should be completed at the highest possible rate. The reaction rate is a function of the temperature, catalyst concentration and liquid phase oxide concentration (which is function of the pressure). The limits on the reactor temperature and catalyst concentration are set by considerations of thermal degradation and purification difficulties. To maximize the reaction rate, the pressure must be kept as high as possible for the entire duration of the batch. The higher limits on the pressure and reaction rate are dictated by the pressure suitability of the reactor system and the feasible heat removal rate.

The mathematical model of the reactor, the heat removal system and the burst-disk orifice, as proposed by Ingham et al (1994), is shown in Table 1. This table is a part of an Excel worksheet, where the model equations, the constant values, the initial values of the variables and the output variable related to each of the model equations are defined and described. Additional explanations are provided for some of the equations, as needed, in the worksheet column entitled “Equation description and explanations”. This part of the worksheet is shown in Table 2. Under normal operating conditions, reacting mass is being re-circulated through the external heat removal system at flow rate of F_c and cooled to temperature T_0 (see equations 3 and 15 in Table 1). The bursting disk is intact ($Open = 0$, see equation 5) and the vapor discharge rate through the orifice, V , is zero (see eq. 7). If for some reason the pressure exceeds the limit of P_{burst} , the burst disk ruptures. In such a case, the variable ‘Open’ becomes greater than zero (eq. 5) and vapor discharge is initiated (eq. 7) at either sonic (eq. 9) or subsonic (eq. 10) discharge rate. The latent heat of vaporization of the discharging oxide cools down the reactor (see eqs. 3 and 13) and the reaction essentially stops. When the disk ruptures, the feed to the reactor is stopped (eq. 6).

It should be noted that this relatively small-scale example is presented since its complete mathematical model can be presented in a table, but much larger scale problems can be tackled using the same techniques.

To run a simulation of the reactor under normal operating conditions, the “Model equations” column, as shown in Table

1, should be saved as an ASCII (Text, OS/2 or MS Dos) file. Polymath can read and execute this file. The results for the temperature and pressure in the reactor under normal operating conditions, for a batch duration of 33 h and 20 minutes, are shown in Figures 3 and 4. It can be seen that the temperature changes are quite moderate. After about 13 h, the temperature reaches a maximal value of 112 °C. The increased reaction rate associated with the high temperature (and pressure) reduces the oxide concentration. This, in turn, affects a reduction of the reaction rate (and the temperature) until the trend is reversed again, showing increasing reactor temperature. The change of pressure presents a similar trend (see Figure 4), where the highest pressure reached, 6.35 bar, is well below the bursting pressure of 8 bar. The molecular weight of the final product in normal operation, is $MW=2895$.

After running the simulation, the initial, minimal, maximal and final values of all the variables as well as plots of some principal variables are copied back to the same worksheet in Excel where the model is stored. This enables comparison of process data with results of normal operation to quickly identify development of any abnormal trends.

Let us consider now a situation where there is a cooling water failure of five minutes duration after 700 minutes (11 h 40 m) from the start of the batch. In order to simulate the reactor’s operation under such conditions, the model description part of the Excel worksheet is copied to a new worksheet, where the necessary changes are introduced into the model equations. One option to simulate the cooling water failure, is to set the re-circulation mass flow rate, F_c at zero. With a small model (as in this Example), the equation whose output variable is F_c , can be easily found (line 25 in Table 1). In large scale problems, the part of the worksheet containing the model equations, output variables definition and description and the rest of the pertinent information, can be defined as a data base. The database management options of Excel enable a rapid search for a variable, change of the associated record, or addition of new records containing new equations for additional output variables, definitions etc. In this particular case the equation on line 25 of the model shown in Table 1, should be replaced by:

$$F_c = \text{if}(t < 700) \text{ then } (3300) \text{ else } (\text{if}(t > 705) \text{ then } (3300) \text{ else } (0))$$

After replacing this equation, the procedure for running the simulation is repeated. The “Model equations” column is saved as an ASCII file and Polymath is used to read the file and integrate the equations. The results for the temperature and pressure in the reactor, in case of cooling failure of five minutes duration, are shown in Figures 5 and 6. It can be seen that the temperature reaches a maximal value of 263.0 °C about 45 minutes after the normal cooling re-circulation rate has been restored. The pressure reaches the threshold value of 8 bar five minutes earlier and the burst disk is ruptured.

Because of the cooling caused by the latent heat of vaporization of the discharging oxide vapor and the reduction of reaction rate to zero due to diminishing oxide concentration, the temperature is reduced to 80 ° C at about two hours after the disk rupture. The pressure is reduced to 1 bar instantaneously. The molecular weight of the product in this case is MW=1325, which is far from the specifications. Thus, this particular batch is lost due to the cooling failure.

There are several strategies that can be used to prevent the temperature runaway and the loss of the batch. One option is to increase the cooling re-circulation rate to its maximum, Fc=5000 kg/min after cooling is restored. To simulate the reactor operation when this option is selected, the equation for Fc must be replaced by:

$F_c = \text{if } (t < 700) \text{ then } (3300) \text{ else } (\text{if } (t > 705) \text{ then } (5000) \text{ else } (0))$

Figure 7 shows the variation of the temperature in the reactor for this case. Indeed, the increased re-circulation rate prevents the temperature runaway. The maximal temperature reached is only 100.5° C and the pressure level stays well below the disk rupture threshold value. Thus, the batch is successfully completed with products of MW=2566, that is slightly below the normal value, but still meeting the specifications.

It should be emphasized that at some stages of the reaction, cooling failures of even fairly long duration, may not result in temperature runaway, thus no intervention is needed to complete a normal batch. However, the right strategy to be used under particular abnormal operational conditions, can only be decided in view of simulation of the reactor operation while accounting for the particular situation under consideration.

5. Conclusions

Simulation of processes under emergency conditions is essential for minimizing damage and preventing a potential disaster. Simulation under such conditions requires a simulator that can be operated effectively by a process engineer, who is often not an expert in numerical methods and computer programming. The simulation may often require changes to be introduced in the mathematical model level (not only in the input of parameters). The analysis of simulation results obtained under emergency conditions may also require access to archive plant data obtained in normal and abnormal operating conditions.

The proposed open architecture simulator uses the Excel spreadsheet as the user interface for input, as a data base management system, as a library of mathematical model components and physical property correlations/constants, and as an archive of process data. Polymath is used as a numerical solver and as the user interface for output. This

combination can be used successfully for the stated purpose. Process engineers will have no difficulty to use Excel as it is most familiar to most of them. Polymath is easy to use with its intuitive modeling language and many options of graphical and tabular presentation of results. The object oriented structure of the model-components library enables easy understanding and modification of the model equations or parameters. Organization of an Excel worksheet as database containing the model equations and key results for a particular scenario, can alleviate and speed-up the analysis of a developing emergency situation.

The flexibility and prompt feedback achieved by the proposed simulator structure have a great potential in process hazard assessment during the actual operation, not just at the design stage.

References

- Cutlip, M. B. and Shacham, M. *Problem Solving In Chemical Engineering*
Prentice-Hall, Upper Saddle River, New-Jersey, 1999
- Dhurjati, P. S., Lamb, D.E. and Chester, D., "Experience in Development of an Expert System for Fault Diagnosis in a Commercial Scale Chemical Process", in Proceedings of the 1st FOCAPO Conference, Ed. Reklaitis, G. V. and Spriggs, H. D. pp. 589-619, Elsevier, New York, 1987.
- Fogler, H. S., *Elements of Chemical Reaction Engineering*, 3rd Ed., Prentice Hall, Upper Saddle River, 1999
- Ingham, J., Dunn, I. J., Heinzle, E. and J. E. Prenosil, *Chemical Engineering Dynamics*, VCH, Weinheim, 1994
- Kneale, M. and G. M. Forster, "An Emergency Condensing System for a Large Propylene Oxide Polymerization Reactor", I. Chem. E. Symp. Series No. 25, 98 (1968)
- Marquardt, W., von Wedel, L. and Bayer, B., Perspectives on Lifecycle Process Modeling", proceedings of the 5th International Conference on Foundations of Computer Aided Process Design, Breckenridge, Colorado, July 18-23, 1999
- Shacham, M., N. Brauner, and M. Pozin, "Application of Feedback Control Principles for Solving Differential-Algebraic Systems of Equations in Process Control Education", *Computers chem. Engng.* 20(Suppl), s1329-s1334 (1996)
- Shacham, M. and M.B. Cutlip, "A Comparison of Six Numerical Software Packages for Educational Use in the Chemical Engineering Curriculum", *Computers in Education Journal*, Vol. IX, No. 3, 9-15 (1999).

Srinivasan, R. and Venkatasubramanian, V., "Multi-perspective Models for Process Hazards Analysis of Large Scale Chemical Processes", Computers chem. Engng., vol 22. Supplement, S961-S964 (1998)

Zaarur, N and M . Shacham, "A layers Architecture Based Process Simulator", proceedings of the 5th International Conference on Foundations of Computer Aided Process Design, Breckenridge, Colorado, July 18-23, 1999

Table 1. Model equations and output variable description for the example

Output variable				
No.	Name	Definition	Initial value	Model equations - Runaway polymerization reaction
1	M	Total mass in the reactor (kg)	M(0)=4400	$d(M)/d(t) = F-V$
2	MC	Oxide mass in the reactor (kg)	MC(0)=0	$d(MC)/d(t) = F-V-r$
3	TR	Temperature in the reactor (° C)	TR(0)=80	$d(TR)/d(t) = (Hc-Hv-Qg-Qr)/(M^*Cp)$
4	X	The mass of oxide reacted (kg)	X(0)=0	$d(X)/d(t) = r$
5	Open	Status of the burst disk: 0 closed, >0 open	Open(0)=0	$d(Open)/d(t) = \text{if } (P < P_{burst}) \text{ then } (0) \text{ else } (0.001)$
6	F	Oxide feed rate (kg/min)		$F = \text{if } (Open > 0) \text{ then } (0) \text{ else } (100)$
7	V	Vapor discharge rate (kg/min)		$V = \text{if } ((P <= 1) \text{ or } (Open == 0)) \text{ then } (0) \text{ else } (V1)$
8	V1	Vapor discharge rate (kg/min)		$V1 = \text{if } (P < 1.9) \text{ then } (V_{subs}) \text{ else } (V_s)$
9	Vs	Sonic vapor discharge rate (kg/min)		$V_s = 0.85 * K_v * P / \sqrt{TR + 273}$
10	Vsubs	Sub-sonic - vapor discharge rate (kg/min)		$V_{subs} = K_v * P / \sqrt{(TR + 273)} * \sqrt{1 + 1/P^2}$
11	r	Reaction rate (kg oxide/min)		$r = k * MC$
12	Hc	Feed enthalpy change (kJ/min)		$Hc = F * Cp * (T_0 - TR)$
13	Hv	Latent heat of vapor discharge (kJ/min)		$Hv = V * \text{Lamda}$
14	Qg	Heat of reaction (kJ/min)		$Qg = r * HR$
15	Qr	Heat removal (kJ/min)		$Qr = F_c * Cp * (TR - T_0)$
16	P	Oxide vapor pressure (bar)		$P = \text{if } (P1 < 1) \text{ then } (1) \text{ else } (P1)$
17	P1	Oxide vapor pressure (bar)		$P1 = (\exp(-3430/(TR + 273)) + 11.7) + 1.45e-3 * MW * C$
18	k	Reaction rate coefficient		$k = 9e9 * \exp(-E/(R * (TR + 273)))$
19	C	Oxide concentration (kg/kg)		$C = MC/M$
20	MW	Molecular weight of the polymer (kg/mol)		$MW = (M_0 + X)/(M_0/74)$
21	T0	Feed temperature (°C)		$T_0 = 80$
22	Lamda	Heat of vaporization of the oxide (kJ/kg)		$\text{Lamda} = 670$
23	Cp	Spec. heat of feed reacting mass (kJ/kg- ° C)		$Cp = 3.5$
24	HR	Heat of reaction (kJ/ kg oxide)		$HR = -1660$
25	Fc	Re-circulation mss flow rate (kg/min)		$F_c = 3300$
26	Pburst	Disk rupture pressure (bar)		$P_{burst} = 8$
27	R	Gas constant		$R = 1.987$
28	E	Activation energy		$E = 21000$
29	M0	Initial alcohol charge (kg)		$M_0 = 4400$
30	Kv	Valve discharge coefficient		$K_v = 100$

Table 2. Model equations, equation descriptions and explanations for the example

Model equations - Runaway polymerization reaction	Equation description, explanations
$d(M)/d(t) = F - V$	Total mass balance
$d(MC)/d(t) = F - V - r$	Oxide comp. Balance
$d(TR)/d(t) = (H_c - H_v - Q_g - Q_r)/(M \cdot C_p)$	Energy balance
$d(X)/d(t) = r$	Reaction rate expression
$d(\text{Open})/d(t) = \text{if } (P < P_{\text{burst}}) \text{ then } (0) \text{ else } (0.001)$	Zero if disk closed becomes >0 when disk bursts
$F = \text{if } (\text{Open} > 0) \text{ then } (0) \text{ else } (100)$	Feed stops when disk bursts
$V = \text{if } ((P <= 1) \text{ or } (\text{Open} == 0)) \text{ then } (0) \text{ else } (V_1)$	No discharge if disk intact or $P > 1$ bar
$V_1 = \text{if } (P < 1.9) \text{ then } (V_{\text{subs}}) \text{ else } (V_s)$	Selection between sub and supersonic discharge
$V_s = 0.85 \cdot K_v \cdot P / \sqrt{TR + 273}$	
$V_{\text{subs}} = K_v \cdot P / \sqrt{(TR + 273)} \cdot \sqrt{1 + 1/P^2}$	
$r = k \cdot MC$	
$H_c = F \cdot C_p \cdot (T_0 - TR)$	
$H_v = V \cdot \text{Lamda}$	
$Q_g = r \cdot HR$	
$Q_r = F_c \cdot C_p \cdot (TR - T_0)$	Heat removed by the heat removal system
$P = \text{if } (P_1 < 1) \text{ then } (1) \text{ else } (P_1)$	$P < 1$ set at 1 to prevent division by zero in V_{subs}
$P_1 = (\exp(-3430/(TR + 273)) + 11.7) + 1.45e-3 \cdot MW \cdot C$	
$k = 9e9 \cdot \exp(-E/(R \cdot (TR + 273)))$	

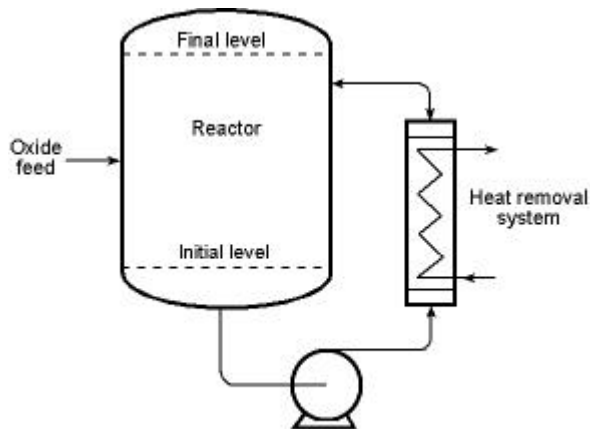


Figure 1. Propylene oxide polymerization reactor (Kneale and Foster, 1968)

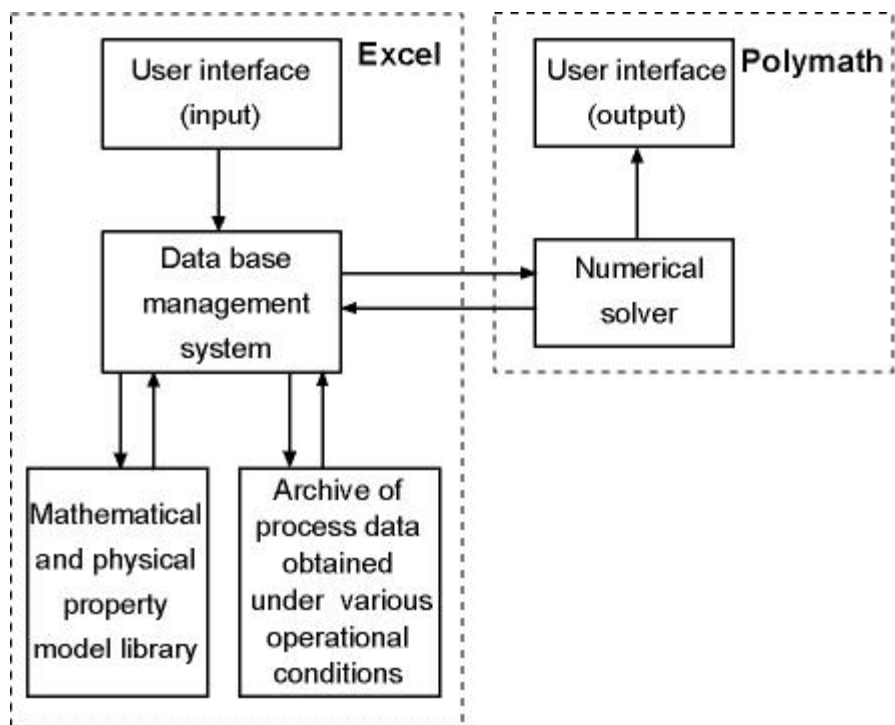


Figure 2. The structure of the process simulator for PHA

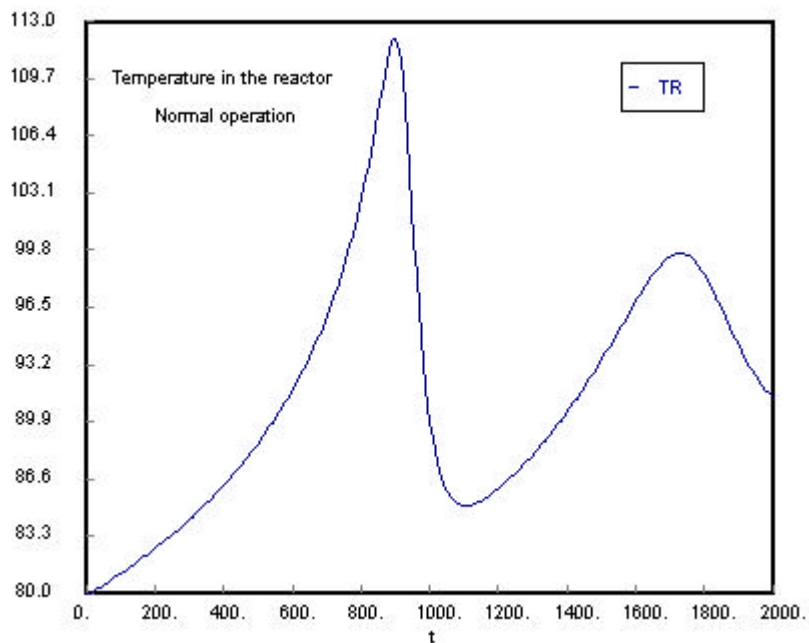


Figure 3. Temperature change in the reactor in normal operating conditions

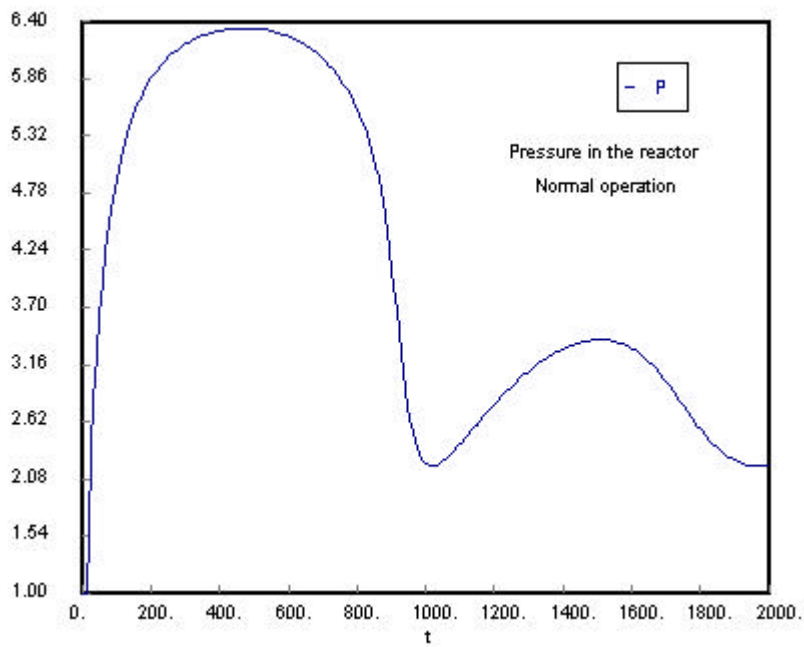


Figure 4. Pressure change in the reactor in normal operating conditions

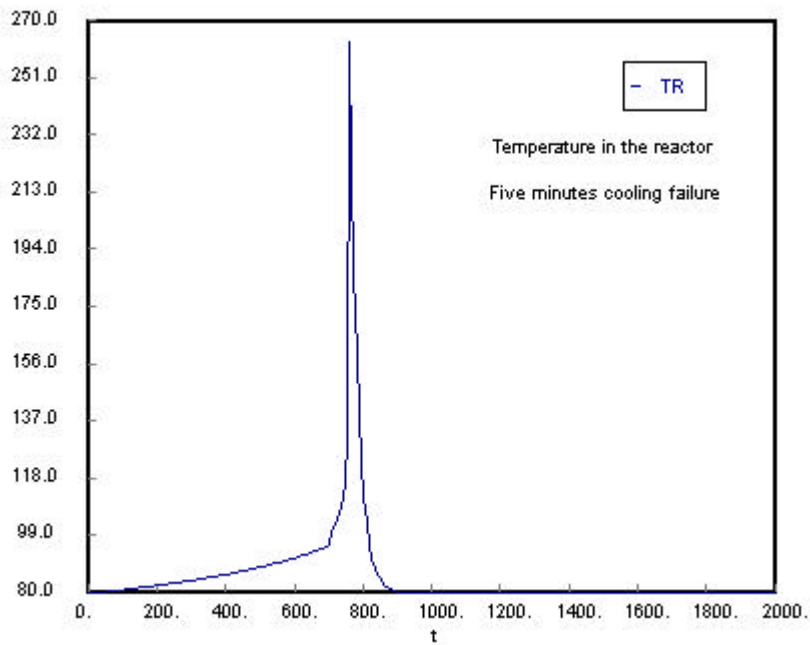


Figure 5. Temperature change in the reactor with five minutes cooling failure.

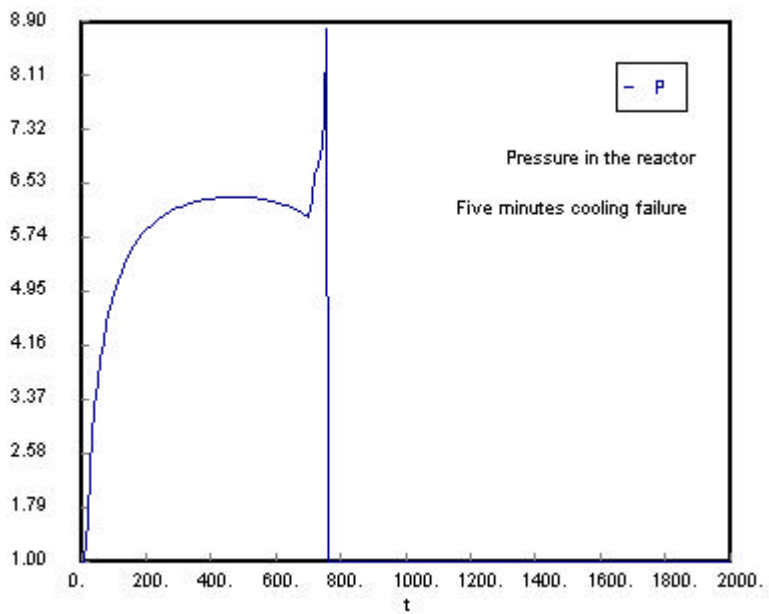


Figure 6. Pressure change in the reactor with five minutes cooling failure

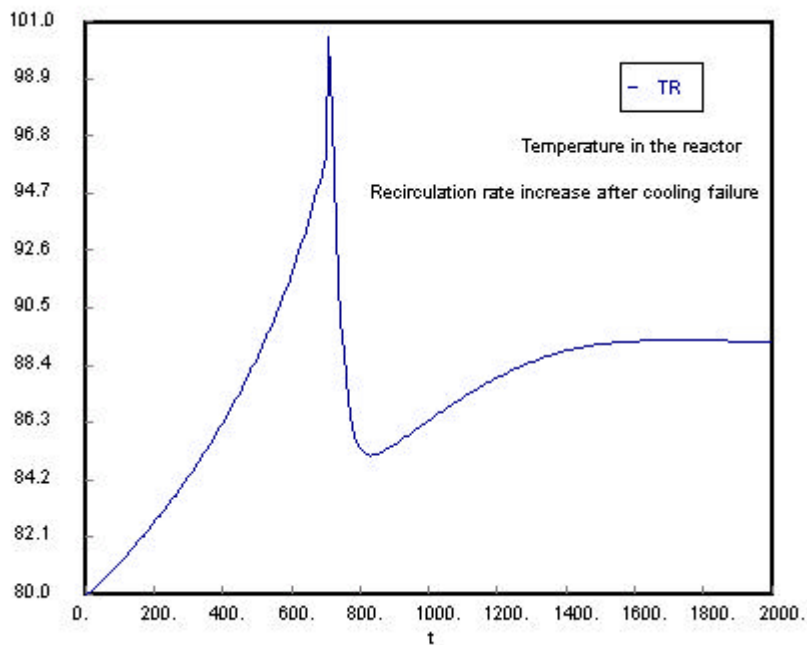


Figure 7. Temperature change in the reactor when re-circulation rate is increased after 5 minutes cooling failure.