Chapter 6

Implementation and Practical Aspects

Nil actum credens, cum quid superesset agendum

6.1 General introduction

The previous chapters have focused on the theoretical basis of the proposed approaches as well on the illustration of the application of such methodologies over case studies with the aim of validate their performance. However, as pointed out in the first chapter, there are some complementary but not less important issues at the time of implementing on-line optimisation systems. These aspects are related to the dynamic and operability associated to the whole system: the plant and the on-line optimisation module. The objective of the current chapter is indeed to address some of such points, which are considered of major significance. The addressed points include considerations about the RTE parameters influence (and controllers) as well software architectures.

6.2 About RTE parameters

6.2.1 Introduction

RTE has been introduced in chapter 3 as a new approach to on-line model-based optimisation. A successful application of such strategy requires an appropriate parameter tuning, that is to say, to answer these questions:

- how often should be the set-points adjusted? and. . .
- what does neighbourhood exactly mean?

This section then describes a proposed procedure for evaluating the influence of the RTE parameters, as well as the required tuning steps. Although the optimal values of these parameters strongly depends on the process dynamics and involves complex calculations, this section uses Williams-Otto benchmark (section 3.3.1, page 57) to obtain general guidelines and illustrates the methodology for the parameter tuning as a function of the process information.

6.2.2 The parameter tuning problem

For a given process and a given set of disturbance patterns entering to the system, the tuning problem consists in finding appropriate values for RTE parameters: time between executions (δt) and the neighbourhood (maximum allowed changes, represented by the vector δx). Following, the influence of such parameters over the system performance is summarised, and some guidelines are extracted to properly adjust them.

6.2.2.1 Neighbourhood size δx

When the improvement procedure is repeated N times (and the disturbances keep constant values), the local optimum is expected to be found with an acceptable degree of accuracy. The greater the values in δx the more inaccurate the result. The lower the values in δx the higher the possibilities of being trapped in a saddle point, being affected by rounding errors (since at every point requires solving a non-linear equation system) and the lower the possibility of reaching the final value within the given number of iterations (N). Thus, δx can be considered as a parameter of an optimisation algorithm which consists in the recursive application of the improvement procedure (section 3.2.2, page 50). Since the optimisation procedure only determines the best steady state, the tuning of δx can be then de-coupled from the process dynamics. In this way, the tuning problem can be stated as finding δx such that the distance between the true optimal point (IOF_{opt}) and that one found using the RTE recursive algorithm (IOF_{RTE}) is minimised for all the expected conditions:

$$\min_{\delta x} \quad Z = \frac{1}{m}\left[\sum_{i=1}^{m}\left|\frac{IOF_{RTE}(\delta x, p_i) - IOF_{opt}(p_i)}{IOF_{opt}(p_i)}\right|^{q}\right]^{\frac{1}{q}} \tag{6.1}$$

where $p_i \ \forall i = 1 \ldots m$, is the discretisation of the range of possible values for the disturbances with economical influence.[1] Obviously, only in few cases is possible to identify the “true” analytical optimal (IOF_{opt}), but it can be approximated by a reference optimisation method able to give the

[1]Note that in the general sense, the equation 6.1 should consider the probability distribution of de disturbance $pdf(p)$, taking then the form:

$$Z = \sqrt{\int \left[\frac{IOF_{RTE}(\delta x, p) - IOF_{opt}(p)}{IOF_{opt}(p)}\right]^2 pdf(p)dp}$$

for the standard quadratic error. However, for the sake of simplicity, the uniform probability density function form has been used.

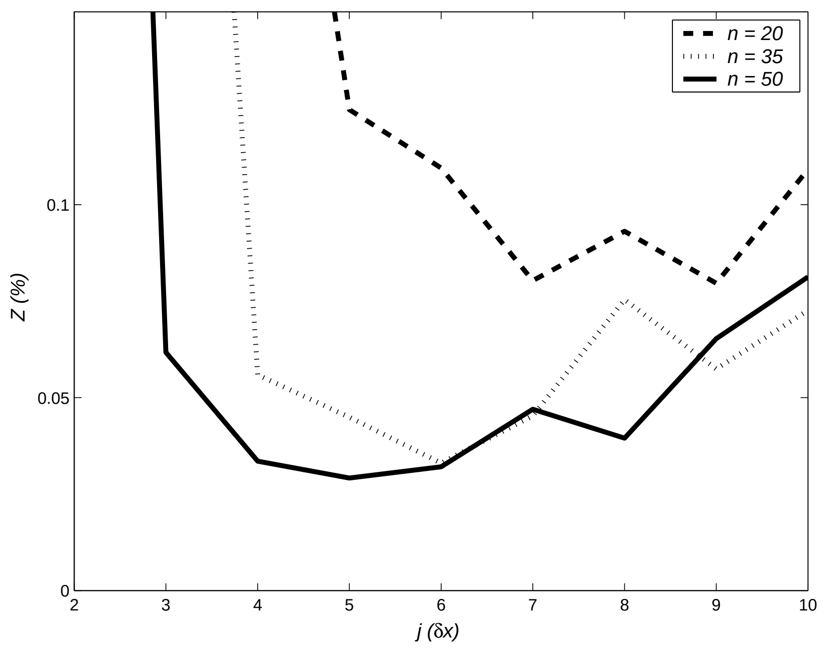


Figure 6.1: Influence of δx on Z and its tuning

optimal with the desired degree of accuracy. Additionally, q will commonly be assigned to two (Euclidean distance). Then, the procedure becomes:

1. Identify the range of variability of the disturbances to evaluate the $p_i \;\; \forall i = 1 \ldots m$ values.
2. Select the reference method for estimating IOF_{opt}.
3. Solve the minimisation problem (equation 6.1).

An appropriate scaling of IOF and the decision variables x will likely allow using the same value for the vector δx components, and thus reducing substantially the computational effort. Figure 6.1 shows the value of Z for increasing values of δx using different and arbitrary values of N. It can be seen that for a changing N there is an acceptable range rather than a punctual optimum (in this case $j_{opt} \approx 5-7$, where δx is proportional to j, an auxiliary integer variable). Such fact is indeed desirable for the overall procedure, as is explained in a subsequent section.

6.2.2.2 Execution period δt

A given disturbance p triggers the RTE procedure that will periodically improve the set points until no further improvement is possible. By changing the RTE frequency (e.g. $\frac{1}{\delta t}$) different set points profiles are obtained for the same disturbance pattern, as it is illustrated in figure 6.2 for the step disturbance case. Consequently, the process performance will different for every δt (even when δx is fixed). For comparisons purposes the scalar MOF can be used (equation 3.11), as it is shown in figure 6.3.

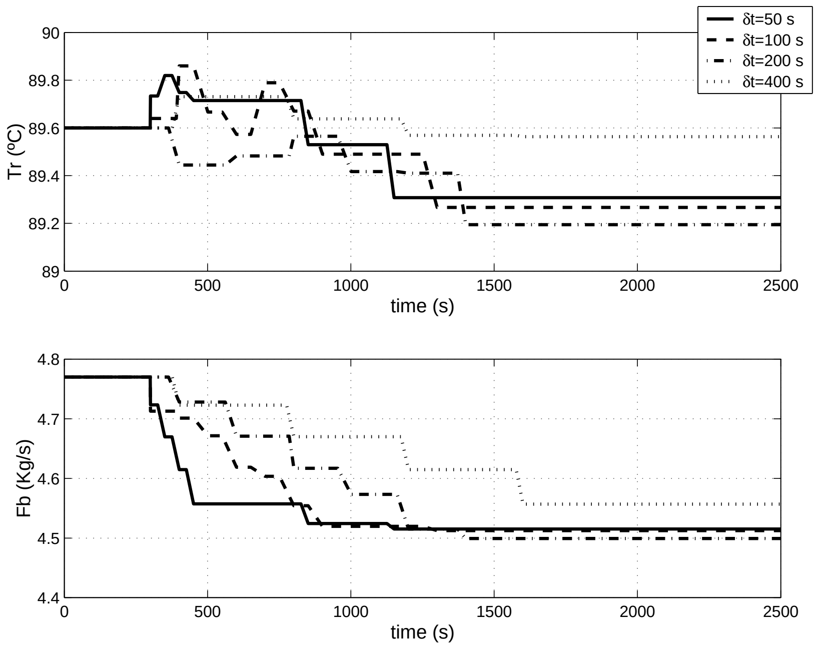


Figure 6.2: Influence of δ*t* over the decision variables profiles for a step disturbance

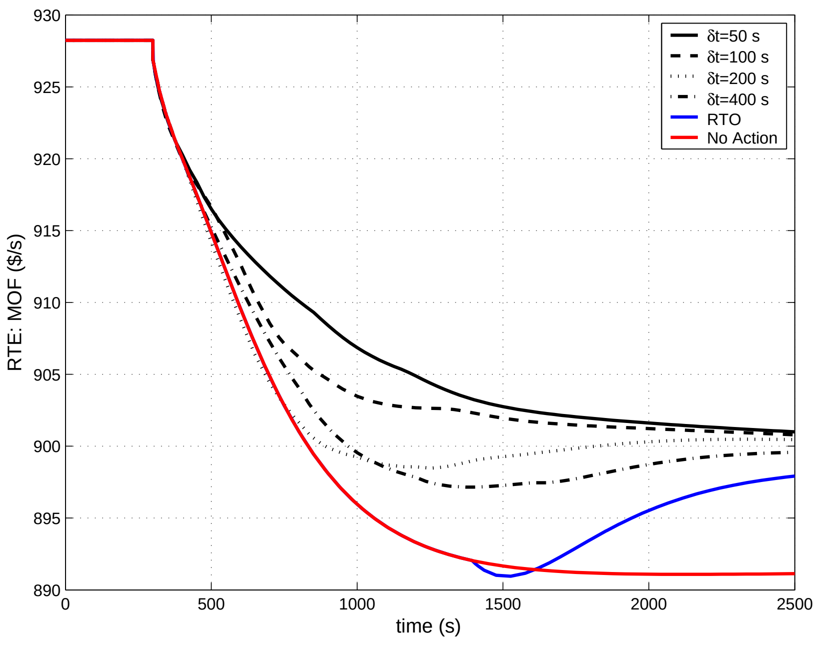


Figure 6.3: Influence of δ*t* over the economical performance for a step disturbance

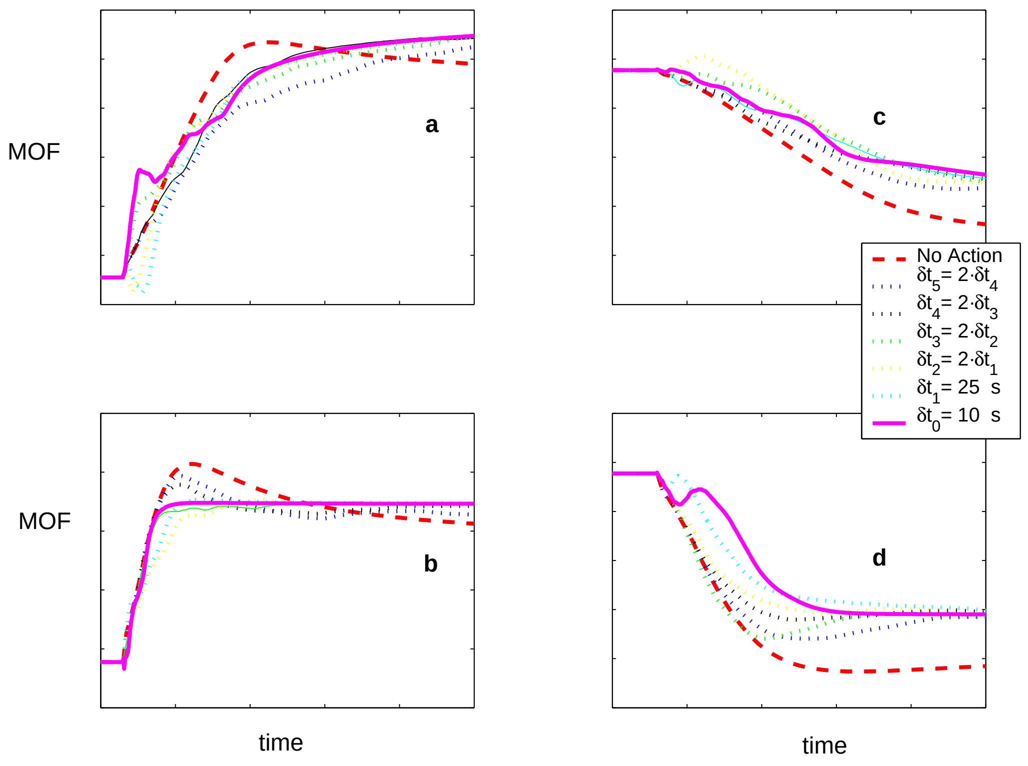


Figure 6.4: *MOF* response of RTE system for different ramp disturbances and the influence of δt

The effect of δt on the system performance has been studied by exciting the system with different ramp disturbances (with the same final values and different slopes, $\frac{dp}{dt}$) and applying RTE with different δt values. Figure 6.4 summarises some of the *MOF* profiles obtained. Charts 6.4-a and 6.4-b correspond to disturbances that favour the steady state objective function, being the opposite for charts 6.4c and 6.4d.

It has been observed, that when the disturbance makes the steady state objective function decrease (in this case when $\frac{dp}{dt} < 0$), the smaller the δt value, the better the performance in terms of *MOF*. There is a point (δt_a) from which the benefit of reducing δt is negligible. Besides, as the slope of the disturbance increase, the δt_a value decreases.

On the other hand, when the disturbance makes the steady state objective function increase (in this case $\frac{dp}{dt} > 0$), the bigger the δt value, the better the performance in terms of *MOF*. There is also a point (δt_a) from which the benefit of increasing δt is negligible. In addition, δt_a value increases with the slope of the disturbance.

Such observation is summarised in figure 6.5, where the shadowed area indicates the region of “good” δt values according to the current value of $\frac{dp}{dt}$. This suggested a short term on-line tuning of δt, (Adaptive RTE, ARTE) according to the current $\frac{dp}{dt}$ value, following for instance the straight dashed-line in figure 6.5.

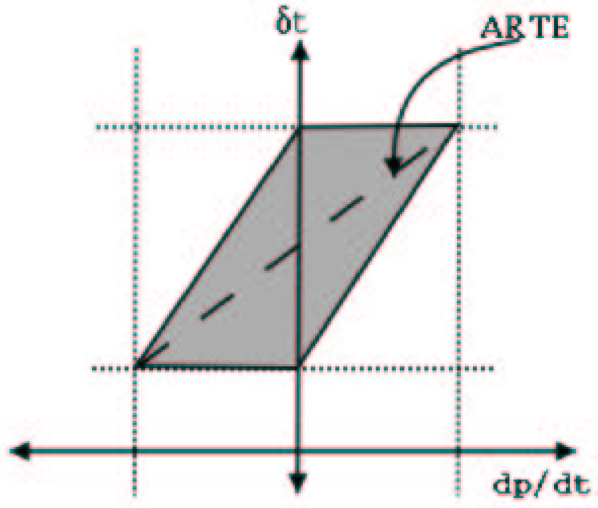


Figure 6.5: Variation of δt_a with $\frac{dp}{dt}$ and potential use of an adaptive RTE approach

Such ARTE policy has been then applied over a long-term simulation for a sinusoidal disturbance. It has been compared with an RTE strategy using different fixed δt values and also with no action as a reference. The results given in figure 6.6 indicate, in opposition to the previous though, that for a persistent disturbance using a fixed δt value works better than the short term ARTE approach, being the latter better only in the small region corresponding to the initiation of the disturbance. That can be explained considering that the *MOF* profiles, although showing relevant information about the performance, are hiding essential information about the capacity of reacting to the next value of the disturbance (the current decision variables' values). Therefore, although during an initial time interval bigger values of δt lead to better performance in terms of *MOF* (figures 6.4-a and 6.4-b), the corresponding process state is not so well prepared for new disturbances as is in the case of lower values of δt. This means that the peaks in figures 6.4-a and 6.4-b in the *No Action* curves correspond just to an inertial effect, which disappears in the case of persistent disturbances.

Looking with more detail the figure 6.6, it can be seen that there is, like in the ramps case, a δt_a value from which further improvement by decreasing δt values is not perceptible. Nevertheless, such δt_a value depends now on the disturbance frequency, rather than its instantaneous derivative. Consequently, an ARTE tuning procedure should produce better results when based in a mid-term and periodical characterisation of the disturbance in frequency terms (e.g. Fourier Transform) instead that using $\frac{dp}{dt}$.[2] Unfortunately, given the non-linearity of the system, a simple linear identification (e.g. using the step response) is not enough appropriate to trust in, at least for this specific case. Therefore, the proposed methodology for the tuning procedure consists in the basic steps depicted in the algorithm 4.

As explained in section 2.6 (page 34), the rate of change of set-points and in consequence of

[2]When several disturbances are present, the dominant one can be used.

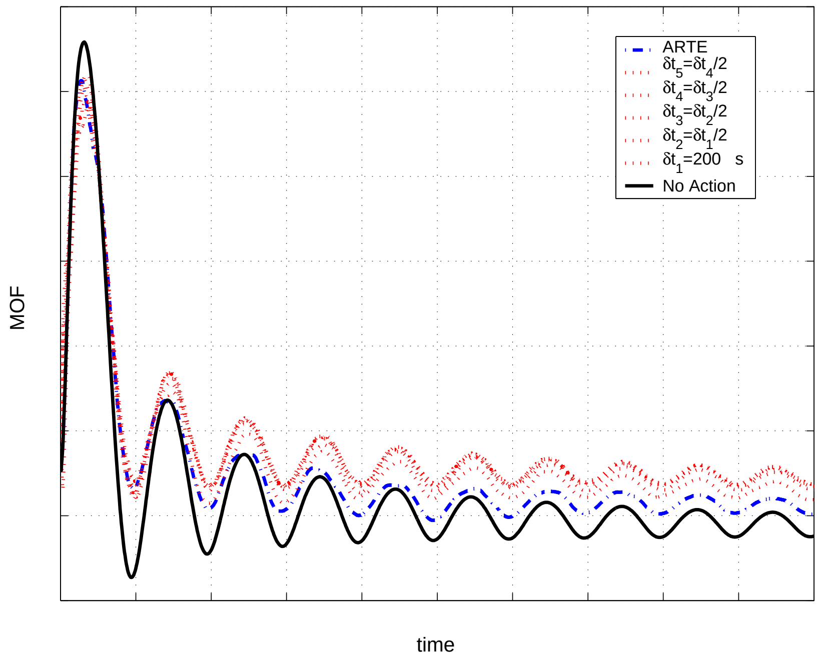


Figure 6.6: System performance for a sinusoidal disturbance

Algorithm 4 Proposed tuning procedure

```
*Initialisation*
Characterise the disturbance in terms of variability
and frequency
Estimate $\delta t_a$
*Main loop*
Do
    Make $\delta t \leftarrow \delta t_a$
    Determine $N$ as $\frac{S_t}{\delta t}$ ($S_t$ is the settling time
     of the process)
    Find the $\delta x$ value that minimises $Z$ (section 6.2.2.1)
    Find $\delta t_a$ (section 6.2.2.2)
Loop Until $\delta t_a \approx \delta t$
```

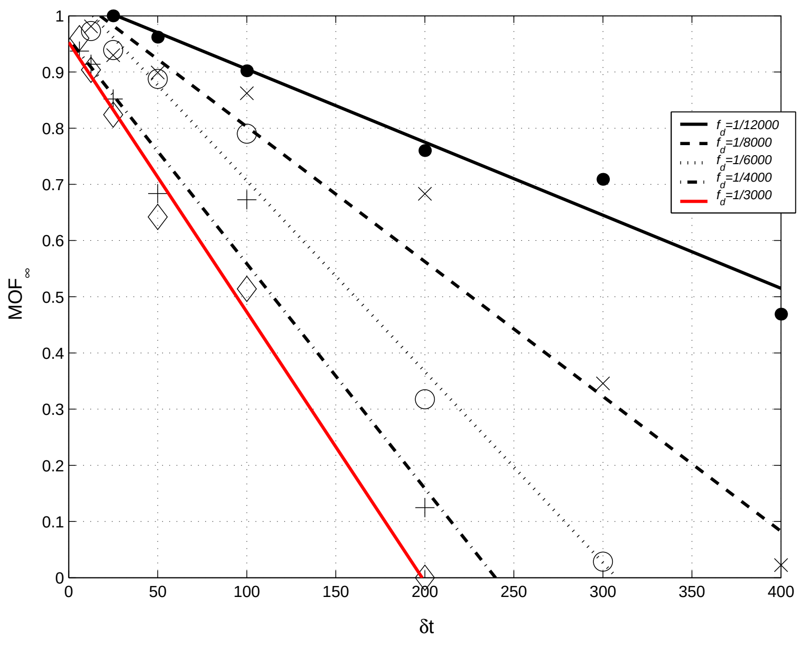


Figure 6.7: MOF_∞ (normalised) for each δt

the decision variables values, $\frac{\delta x}{\delta t}$, must not exceed the capabilities of the control system. Therefore, if the $\frac{\delta x}{\delta t}$ that results applying the proposed procedure do not satisfy such requirement, either δx is decreased or δt increased to allow a safe closed loop operation.

6.2.3 About the influence of controllers on the performance

Figure 6.6 shows how as the time grows, the value of MOF is attenuated because of the damping effect of the integral. In the limit, when the time tends to infinity, MOF will take a finite value, denoted by MOF_∞. It is also observed that the lower the δt values, the higher such MOF_∞ values. If the latter are plotted against δt, an approximately straight line is obtained. In fact, figure 6.7 shows such lines obtained for different disturbances frequencies over the Williams-Otto reactor case study. Such figure is just revealing that for a given frequency of the disturbances, the faster the RTE system the bigger the long-term economical performance.

However, such results corresponds to the extreme case of a perfectly controlled system. If the controllers dynamics are included, a slightly different results is obtained. With the aim of illustrating the controllers influence, consider again the Williams-Otto scenario. The reactor level is then controlled manipulating the outlet flow, F_r. This fact is simulated by on the one hand, properly modifying the dynamic model making $\frac{d(V_r \cdot \rho)}{dt} \neq 0$ in the global balance (equation B.11 of appendix B) and including the controller law as an additional equation:

$$\begin{aligned} F_r = F_r^{bias} + k_p^c e^c + \tfrac{\int e^c dt}{T_i^c} \\ e^c = V_r^{pv} - V_r^{sp} \end{aligned} \tag{6.2}$$

where:

k_p^c:	proportional effect parameter, equal to $0.03\,\frac{kg}{L\cdot s}$.
T_i^c:	integral effect parameter, equal to $500\,\frac{L\cdot s^2}{kg}$.
F_r^{bias}:	bias, equal to $6.6\,kg/s$.
V_r^{pv} andV_r^{sp}:	are the reactor volume and its set-point ($2104\,L$) respectively.

On the other hand, the dynamics of the temperature controller are simulated as a first order lag plus a pure delay, represented in the Laplace variable domain (s) by the following equation:[3]

$$\frac{T_r(s)}{T_r^{sp}(s)} = \frac{e^{-\theta_d s}}{\tau_l s + 1} \tag{6.3}$$

where:

T_r:	is the actual reactor temperature (°C).
T_r^{sp}:	is the set-point for the reactor temperature (°C).
θ_d:	is the pure delay constant (equal to $38\,s$).
τ_l:	is the first order time constant (equal $255\,s$).

When the resulting system model is perturbed using a step disturbance in the F_a flow, a set of profiles are obtained, as those shown in figure 6.8, and in consequence the economical performance is also affected by the controllers (see figure 6.9).

Thus the previous analysis concerning the economical performance (MOF_∞) for the controlled case leads to results slightly different as a consequence of the delay introduced by the control system. The combined system controllers-plant is not so fast like the perfect control case, as is illustrated in figure 6.10. Therefore, the more the control system approximates to the ideal perfect case, the better the economical performance of the RTE implementation.

Given that the control layer does affect the RTE economical performance, another question naturally arrises: does RTE influence the control performance? The following subsection describes a qualitative analysis about this aspect.

[3]Such simplification allows to avoid the energy balance without compromising the validity of the results.

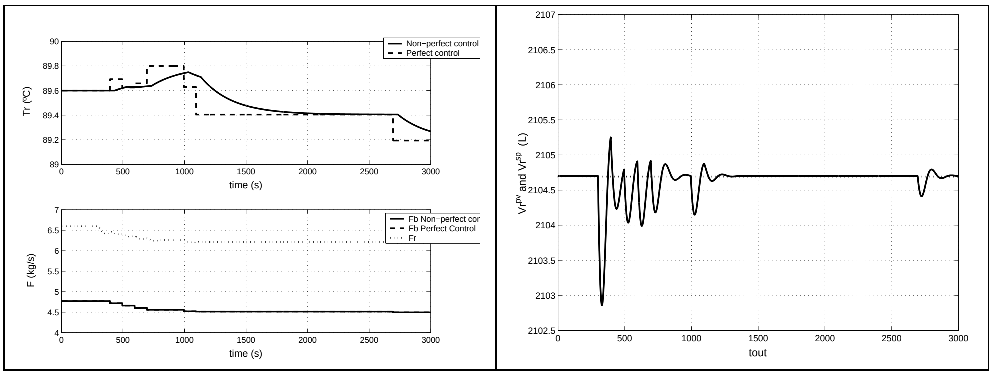


Figure 6.8: Profiles obtained for F_b, F_r, T_r and V_r when the controllers dynamics are not neglected (step disturbance)

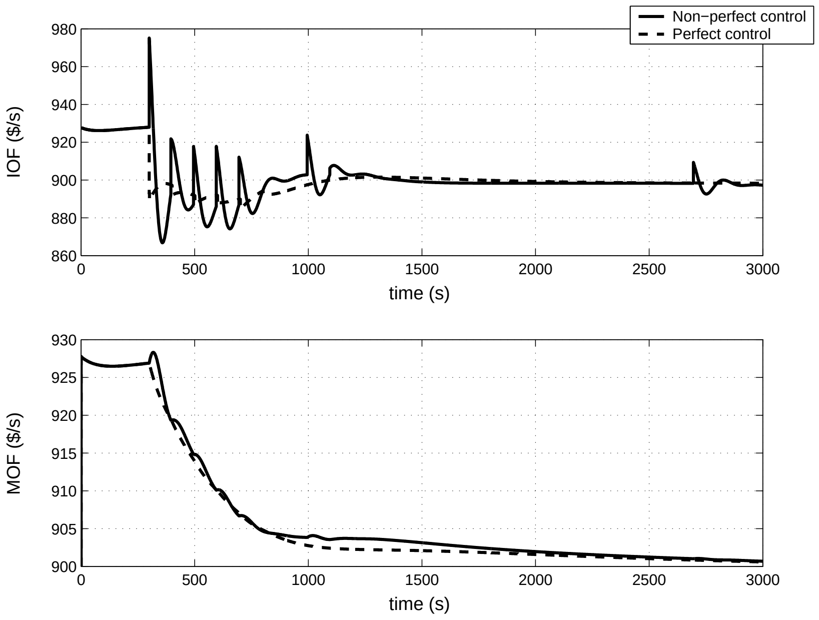


Figure 6.9: *IOF* and *MOF* profiles when the control dynamics are not neglected (for a step disturbance)

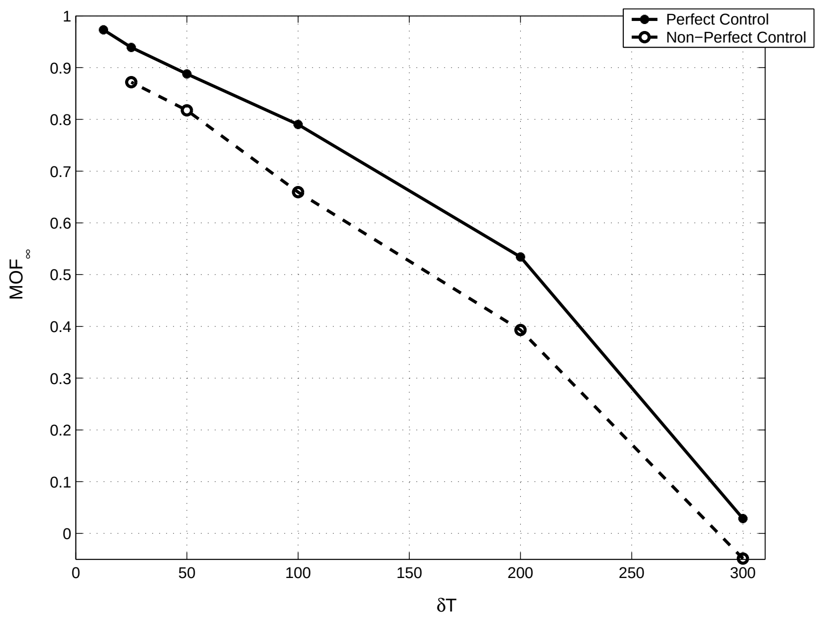


Figure 6.10: Economical influence of the controllers

6.2.3.1 Process stability

Consider then, for illustration purposes, the general simplified scheme of a process and with a feedback control scheme (figures 6.11 and 6.12 respectively). Under certain conditions of linearity, it can be shown that:

$$y(s) = G_{SP}(s)SP(s) + G_{DC}(s)D(s) \tag{6.4}$$

where:

$$G_{SP}(s) = \frac{G_P(s)G_C(s)}{1 + G_P(s)G_C(s)} \tag{6.5}$$

$$G_{DC}(s) = \frac{G_D(s)}{1 + G_P(s)G_C(s)} \tag{6.6}$$

being:

s:	Laplace domain variable.
$y(s)$:	system output.
$SP(s)$:	set-points.
$D(s)$:	disturbances.
$G_P(s)$:	process transfer function.

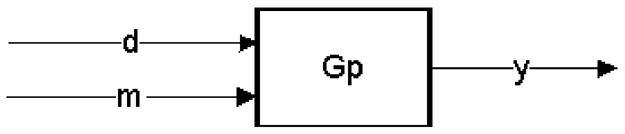


Figure 6.11: Block diagram for a process

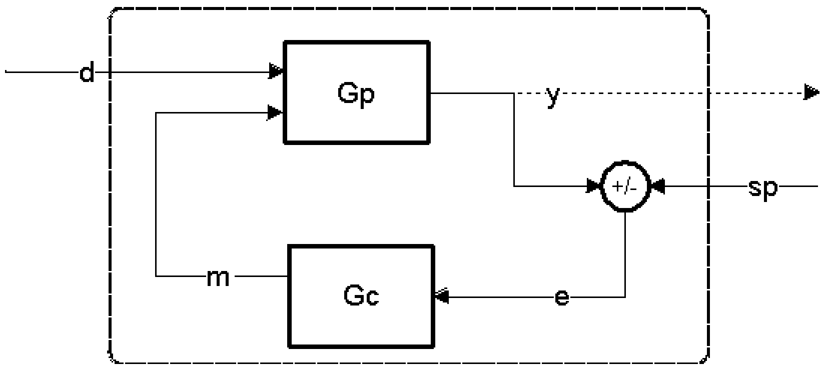


Figure 6.12: Block diagram for a process under feedback control

$G_C(s)$:	controller transfer function.
$G_{SP}(s)$:	transfer function of the closed loop system with respect to the set-point.
$G_{DC}(s)$:	transfer function of the closed loop system with respect to the disturbance.

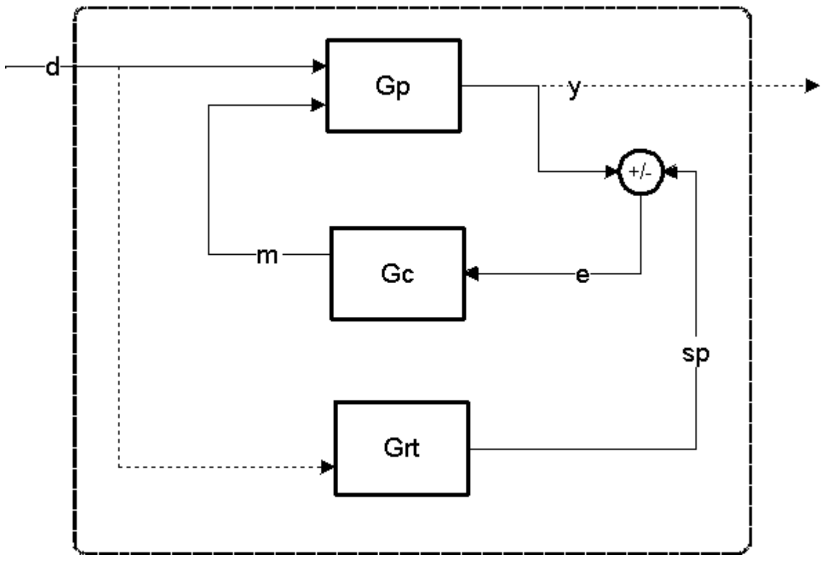


Figure 6.13: Block diagram for a process under feedback control coupled with an on-line optimisation system

Then, when the RTE block is included in the system, the corresponding block diagram becomes like the one indicated in figure 6.13. In such context, the stability of the controlled system can be analysed by properly identifying the roots of the characteristic equation of the whole system highlighted by the dashed lines. As already explained, a RT system obtains *SP* according to the current steady state plant model and disturbance measures. Then defining:

$$G_{RT} = \frac{SP(s)}{D(s)} \tag{6.7}$$

Allows to obtain the following transfer function for the closed loop system:

$$y(s) = [G_{SP}(s)G_{RT}(s) + G_{DC}]\,D(s) \tag{6.8}$$

or equivalently:

$$y(s) = \left[\frac{G_P(s)G_C(s)G_{RT}(s)}{1+G_P(s)G_C(s)} + \frac{G_D(s)}{1+G_P(s)G_C(s)}\right] D(s) \tag{6.9}$$

From the equation 6.9, it can be seen than despite the specific transfer functions of process, controller and RT system, and in accordance with the previous assumptions, the $G_{RT}(s)$ term does not influence the denominator roots, and thus, the closed loop stability. It should be noted, however, that the assumption of linearity is in general terms not valid, but may be acceptable for small movements around a given work point. In any case, the above reasoning reflects the intuitive fact that there is no feedback using the RTE system, since it uses (as RTO does) a rather feed-forward configuration. Nevertheless, strictly speaking there is certain information feedback. Such feedback is produced during the model updating stage, when the steady state is detected and the model parameters are fitted to properly represent latest plant situation. Since such feedback will likely be produced when the plant is stable (since it is at steady state) that fact should not represent a stability problem, but allows to foresee that a too permissive steady state detector constitutes a potential source of stability problems.

6.2.4 Conclusions

This section briefly shows some findings about the influence of RTE parameters on the process economical performance. As a result, a methodology for an adequate tuning of such parameters is proposed. It is shown how the parameters related to control variables can be tuned just by using the steady state model and information about the variability of the disturbances. On the other hand, the time parameter needs both, the characterisation of the disturbance in terms of amplitude and frequency and a further testing over a dynamic simulation of the process. In addition, a periodical characterisation of the disturbances is expected to allow an on-line adaptation of the parameters.

6.3 About software architectures

In section 1.4.2 (page 12) has been argued that some typical causes of unsatisfactory results of on-line optimisation projects are related with the system (software) design and maintenance. The contribution of this thesis to solve such aspect is not determinant, since it is not oriented to develop a software package for on-line optimisation. Nevertheless, software related issues have been not neglected, and although the prototypes developed and implemented in this thesis work for performing the desired functionality are not conceived for its industrial use, they stablish very useful guidelines for a successful software design.

Two main issues are then considered in relation with software design.

- The first one is concerned with the optimisation framework, particularly for the operation optimisation of continuous processes. Such optimisation problems may not necessarily involve many decision variables, but the used/available models may indeed be very complex.
- Secondly, the simulation/optimisation framework of process with decaying performance. As such approach require the implementation of discrete decisions (i.e. clean, operate), special considerations are required, not only for simulation purposes, but also for a appropriate industrial application.

These aspects are explained and exemplified in the following subsections.

6.3.1 Simulation and optimisation framework for continuous processes

6.3.1.1 Introduction

Lots of different software tools are currently available for managing some of the on-line optimisation features and goals. However, most of these tools are presently used for different activities in an independent way. Hence, the opportunity of using these software tools within an integrated environment can be investigated in order to enhance their performance in a synergetic way, taking advantage of the developments in each specific area and combining them. This section introduces a starting point in this direction that is based on the use of commercial software tools (such as simulation and optimisation packages, as well as spreadsheets, which are very familiar to process engineers). Such tools are later integrated into a distributed system with the aid of the presently available communication technologies (COM, DDE). The basic static and dynamic structures are next introduced.

6.3.1.2 System architecture

Optimisation is the core and main functionality of on-line optimisation systems. However, the tasks the system is also expected to manage the data coming from and going back to the plant (directly or indirectly). Hence, the following modules are proposed:

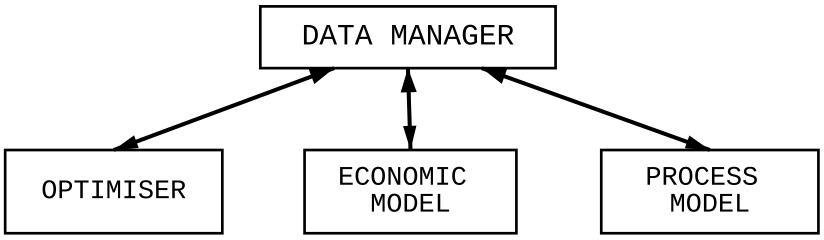


Figure 6.14: Static architecture

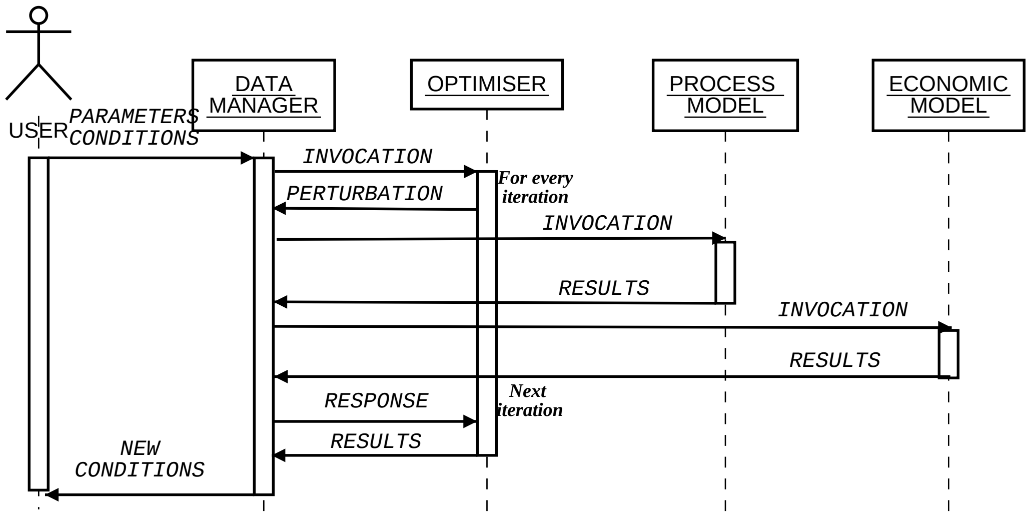


Figure 6.15: Dynamic architecture

Data Manager: this module receives data from the plant (that have previously passed the data improvement sequence explained in section 2.3, page 23). After optimisation this module sends back the set points to the control system. An additionally desirable functionality is that allows an easy off-line analysis.

Optimisation Manager: this module involves the three remaining elements. This elements are the *Process Model* (constraints), the *Economic Model* (objective function) and the *Optimiser* (solver).

The associated static structure is shown in figure 6.14. That is to say, the *Data Manager* is the system's executive, while the remaining elements are interfaced to the *Data Manager* but not between themselves. The double arrow lines in figure 6.14 show that the communication is reciprocal. The process and economic models are basically black box simulators, in the sense that they produce an answer when perturbed by an input, under certain model parameter conditions. On other hand, figure 6.15 gives the corresponding dynamic structure.

Following the scheme in figure 6.15, as the *Data Manager* receives the new plant conditions, the solver (*Optimiser*) is invoked for optimisation. The *Optimiser* will then perturb the model and wait for response the times needed for the corresponding algorithm to converge. This is a non-direct perturbation since the data are sent to the *Process Model* trough the *Data Manager*. Once the *Process Model* gets the results, these results are sent to the *Economic Model* again trough the *Data Manager*.

Off-line analysis is also possible within the same structure. The *Optimiser* may be replaced or complemented by any other module (e.g. in particular, the improvement algorithm for RTE) since the only requirement for the new component to be plugged into the system is to have the same dynamic behaviour as the *Optimiser* module. Thanks to this architecture, the use of different algorithms and modules is extremely easy and helpful. Sensitivity analysis is an example of off-line process that may be included in this framework as it will be shown later on.

6.3.1.3 Implementation

A simple framework has been implemented according to the previous scheme, using two illustrative cases (A and B). For the purposes of this thesis work, the software election for the figure 6.14 functionality is following given:

The *Process Model* has been developed using the sequential-modular simulation package HYSYS.Plant since it offers very good communication capabilities. On other hand, the *Economic Model* has been implemented on a MSExcel's spreadsheet. Excel also has an open architecture allowing very easy communication with HYSYS via ActiveX. Additionally, it is a tool that may be considered as a customisable user interface for the system. The economic model implemented includes the investment calculation, the operating cost, and the revenues.

In Case A, given that the optimisation involves only one decision variable, the *Optimiser* was programmed in Visual Basic for Applications (VBA), a simple language, easily linked with Excel. The algorithm used has been the golden section because of its robustness. On other hand, in Case B as the optimisation involves more decision variables no algorithm was written. Instead, a function of the Matlab's Optimisation Toolbox (a SQP implementation) has been used. Other solvers (public domain, in house, etc.) may be also plugged in this scheme thanks to the modular approach.

The *Data Manager* chosen for this specific implementation is Excel again. It allows the easier communication according the previous choices, and this is the main job of the *Data Manager* (Rosen and Partin, 2000).

6.3.1.4 Communication

Given all the elements of the system, communication between them is needed. Regarding the tools selected for the specific implementation presented, the simplest way is to use VBA and the interfaces available for Excel and HYSYS. In order to communicate the algorithm in Matlab with the *Data Manager* in Excel, the Dynamic Data Exchange (DDE) service was used.

6.3.1.5 Some examples for validation

For evaluation of the system architecture two very simple but demonstrative scenarios are presented. In the first case study, a typical plant scheme is considered: a plant with a reactor, a separator and a recycle stream. In this case the single decision variable is the recycle flow and the objective function is the plant profit. For the second example more decision variables are contemplated. Both cases are from (Seider et al., 1999).

Case A: the problem. The process under study is the ethyl chloride production according to the process flowsheet shown in figure 6.16. A feed stream containing HCl, C_2H_4 and N_2 is mixed with a recycle stream and send to a conversion reactor where the following reaction takes place:

$$\overbrace{C_2H_4}^{ethylene} + \overbrace{HCl}^{hydrogen\,chloride} \rightarrow \overbrace{C_2H_5Cl}^{ethyl\,chloride}$$

The reactor outlet is then separated into a product stream (P) and the stream (S4) to be recycled. A purge stream (W) prevents the accumulation of inert components (N_2) in the system. Hence, changes on W flow affect the recycle and thus the process economy. Increasing the recycle increases energy cost, but raw materials cost is reduced instead. Therefore, the flow W constitutes a decision variable, and should be adjusted to produce the best operation point according to the changing feed conditions, market prices and resources cost.

Case A: results. System performance is verified varying key model inputs, like parameters, the feed conditions and the market prices. When input changes occur, the new model response (simulation according to the new plant data) and the last model conditions are compared. If the changes are significant, the optimisation takes place. After optimisation is done, the *Data Manager* exposes the new set-points to the control system.

Figure 6.17 (left) shows a screen-shot of the *Data Manager*'s main page. Row divisions show to the different input/output sets, input parameters, manipulated variable, objective function and main process model outputs. The first column is for the variable name, next column (Plant) contains the plant data and the simulation results obtained from those data. The last column (Last Model) gives the last optimisation conditions and results. Finally, the centre column gives the difference between the previous values, used for deciding if next optimisation will be done.

For time consumption evaluation, random feed conditions were generated for ten scenarios and the average optimisation time was 23.4 seconds (std. deviation 23.5 %) in a PC Pentium Family AT 350 MHz - Ram 128 KB. This is similar to the time required by the HYSYS optimiser, but this approach allows in addition to deal with a more open and flexible model.

The same framework may offer other decision-making tools such as off-line sensitivity analysis. Figure 6.17 (right) shows a screen-shot of the page used for this purpose. The chart shows the effect of the W stream flow on the Present Value . This kind of study can be very useful when the calculated optimum can not be achieved for practical reasons not contemplated in the model

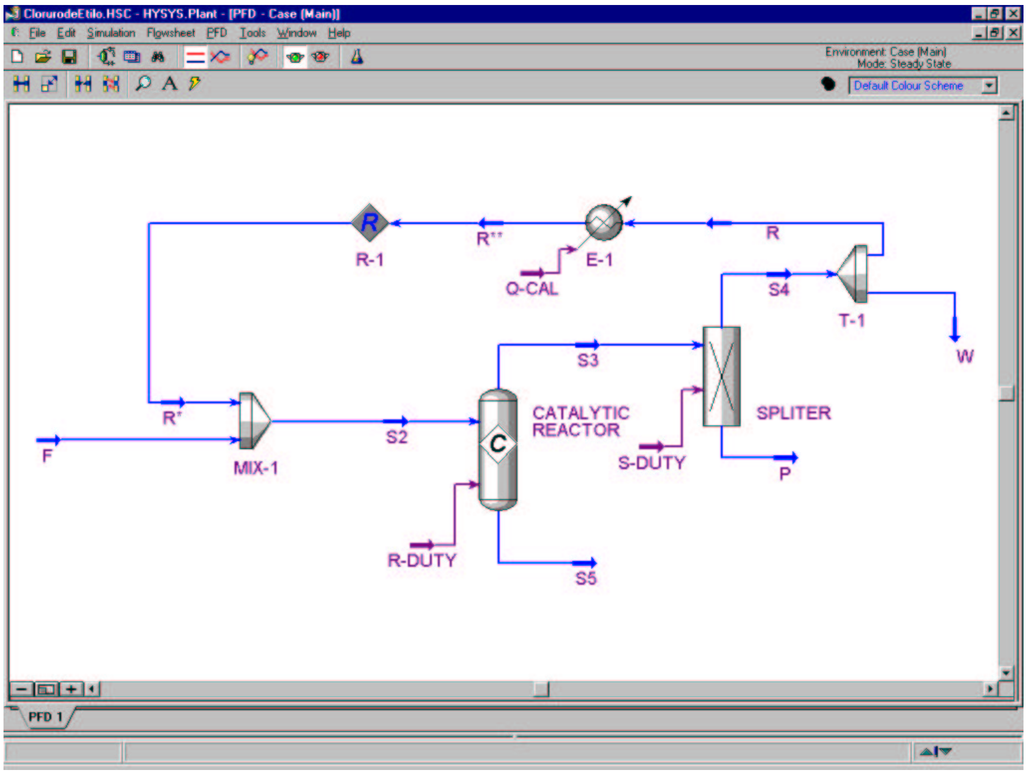
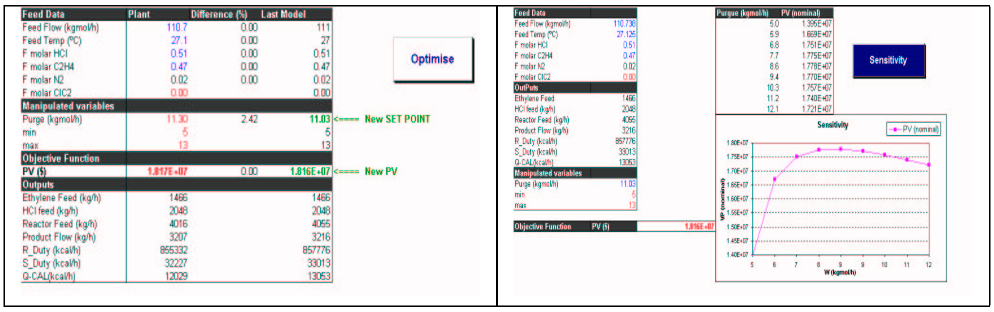


Figure 6.16: Ethyl-chloride production process flowsheet



Feed Data	Plant	Difference (%)	Last Model
Feed Flow (kgmol/h)	110.7	0.00	111
Feed Temp (ºC)	27.1	0.00	27
F molar HCl	0.51	0.00	0.51
F molar C2H4	0.47	0.00	0.47
F molar N2	0.02	0.00	0.02
F molar ClC2	0.00		0.00
Manipulated variables			
Purge (kgmol/h)	11.30	2.42	11.03 <==== New SET POINT
min	5		5
max	13		13
Objective Function			
PV ($)	1.817E+07	0.00	1.816E+07 <==== New PV
Outputs			
Ethylene Feed (kg/h)	1466		1466
HCl feed (kg/h)	2048		2048
Reactor Feed (kg/h)	4016		4055
Product Flow (kg/h)	3207		3216
R_Duty (kcal/h)	855332		857776
S_Duty (kcal/h)	32227		33013
Q-CAL(kcal/h)	12029		13053

Feed Data	
Feed Flow (kgmol/h)	110.738
Feed Temp (ºC)	27.125
F molar HCl	0.51
F molar C2H4	0.47
F molar N2	0.02
F molar ClC2	0.00
OutPuts	
Ethylene Feed	1466
HCl feed (kg/h)	2048
Reactor Feed (kg/h)	4055
Product Flow (kg/h)	3216
R_Duty (kcal/h)	857776
S_Duty (kcal/h)	33013
Q-CAL(kcal/h)	13053
Manipulated variables	
Purge (kgmol/h)	11.03
min	5
max	13

Objective Function	PV ($)	1.816E+07

Purgue (kgmol/h)	PV (nominal)
5.0	1.395E+07
5.9	1.669E+07
6.8	1.751E+07
7.7	1.775E+07
8.6	1.778E+07
9.4	1.770E+07
10.3	1.757E+07
11.2	1.740E+07
12.1	1.721E+07

Figure 6.17: Left: *Data Manager*'s main page. Right: Page for sensitivity studies

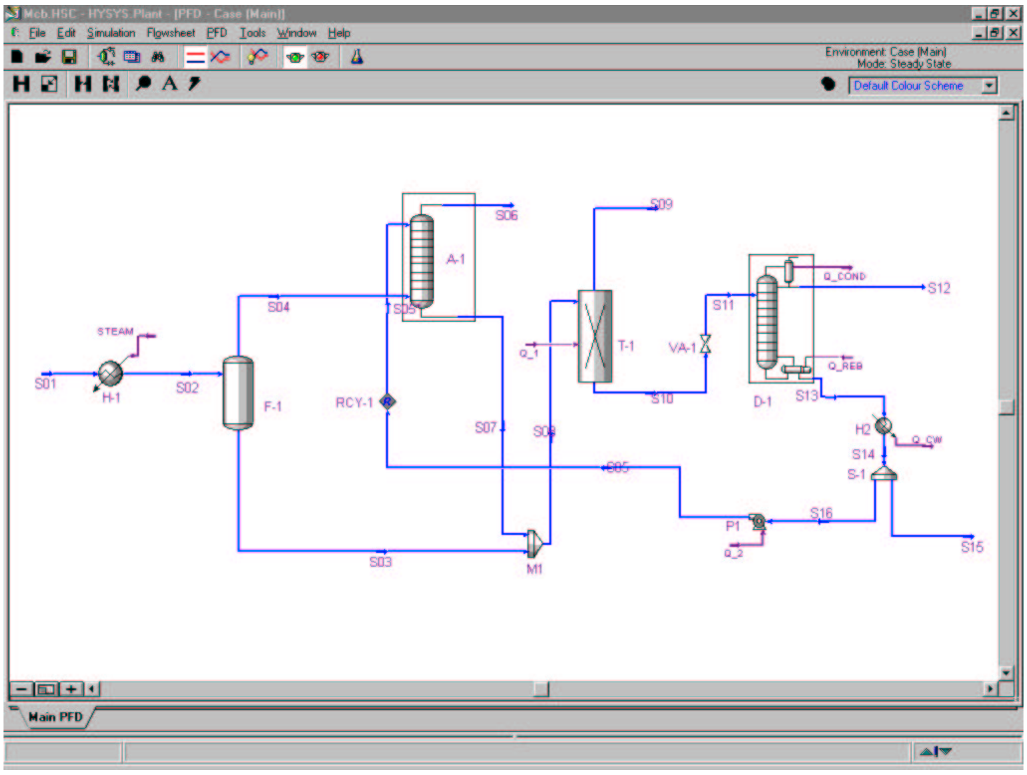
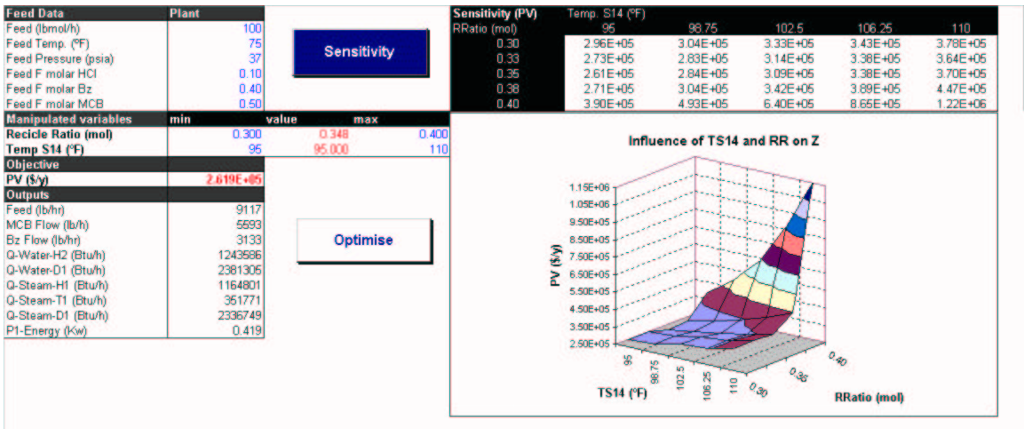


Figure 6.18: Bz-MCB separation process flowsheet

and for bottleneck identification as well.

Case B: the problem. The system considered next is the separation of a mixture of hydrogen chloride (HCl), benzene (Bz), and mono-chloro-benzene (MCB), which is the output of a reactor producing MCB by the chlorination of benzene. The process flowsheet is shown in figure 6.18.

Consider the influence of temperature in stream S14, which is given by MCB final cooling. A temperature increase will produce on the one hand a better Bz recovery, but on the other hand will increase the refrigeration cost. Another interesting trade-off is given by the split fraction of



Feed Data	Plant		
Feed (lbmol/h)	100		
Feed Temp. (ºF)	75		
Feed Pressure (psia)	37		
Feed F molar HCl	0.10		
Feed F molar Bz	0.40		
Feed F molar MCB	0.50		
Manipulated variables	**min**	**value**	**max**
Recicle Ratio (mol)	0.300	0.348	0.400
Temp S14 (ºF)	95	95.000	110
Objective			
PV ($/y)	2.619E+05		
Outputs			
Feed (lb/hr)	9117		
MCB Flow (lb/h)	5593		
Bz Flow (lb/hr)	3133		
Q-Water-H2 (Btu/h)	1243586		
Q-Water-D1 (Btu/h)	2381305		
Q-Steam-H1 (Btu/h)	1164801		
Q-Steam-T1 (Btu/h)	351771		
Q-Steam-D1 (Btu/h)	2336749		
P1-Energy (Kw)	0.419		

Sensitivity (PV)	Temp. S14 (ºF)				
RRatio (mol)	95	98.75	102.5	106.25	110
0.30	2.96E+05	3.04E+05	3.33E+05	3.43E+05	3.78E+05
0.33	2.73E+05	2.83E+05	3.14E+05	3.38E+05	3.64E+05
0.35	2.61E+05	2.84E+05	3.09E+05	3.38E+05	3.70E+05
0.38	2.71E+05	3.04E+05	3.42E+05	3.89E+05	4.47E+05
0.40	3.90E+05	4.93E+05	6.40E+05	8.65E+05	1.22E+06

Figure 6.19: Page for sensitivity studies

S14 in separator S-1. As the recycle grows, more Bz will be again recovered, but the energy cost is increased (heating in D-1 and T-1). Finally, another operation decision variable is the Reflux Ratio in tower D-1, which may be adjusted to satisfy the flow and purity of products directly in HYSYS. Hence the problem arising is the determination of the optimum values of these variables once that feed conditions and market prices are given.

Case B: results. Once again, the system performance is tested varying input conditions for the model such as feed attributes and market prices. The *Data Manager* structure is the same showed in Case A. Figure 6.19 shows a screen-shot of the page used for sensitivity studies. The chart shows the effect of the decision variables on the present value: the strong effect of the temperature, and the minor effect of the split fraction. The required time for performing optimisation has been measured using ten scenarios for which random feed conditions were generated. The average optimisation time was 66.7 seconds (std. deviation 19.5 %) in a PC 350 MHz - Ram 128 KB, which is of the same order than that required by HYSYS optimisers. However, the modular approach also allows using different optimisation techniques, which may improve the performance by manipulating the aggregated model. The recycle stream for instance, may be considered at an upper level by adding new decision variables and new constraints. This is, the recycle stream may be ignored in HYSYS, while the recycle stream may be considered as manipulated variable. Hence, by adding the recycle convergence as a set of constrains, the solver can optimise the process while converging the flowsheet. Thus, there is no need to wait HYSYS to converge the recycle for each solver iteration. Using this infeasible path strategy on ten scenarios, the average optimisation time was 132.5 seconds (std. deviation 10.3 %) on the same PC, which is certainly higher than for the original model. However, it will likely not be the case as more decision variables are implicated.

6.3.1.6 Final comments

A simple framework for the development and implementation of the essential elements of an on-line optimisation system has been introduced. The advantages of the proposed architecture are that known tools for the chemical engineer are the main elements used. Besides, the engineer effort is mainly focused on the system communication and the decision-making, but not in the development of its constitutive parts. Experience and development in the areas of optimisation and simulation may be included in the system by communicating the corresponding software packages. Additionally, the modular approach allows the use of plug and play philosophy. This means easily construction and maintenance, one of the key points for software success (see section 1.4.2, page 12). Indeed, the benefits of the proposed structure has been also manifested when being applied over the continuous process scenarios early presented in chapters 3 and 5 (i.e. the PROCEL and the Paraffins Separation Plant scenarios). Finally, the communication efforts are expected to be significantly reduced after standardisation processes (i.e. CAPE-Open standards).[4]

6.3.2 Simulation and optimisation framework for process with decaying performance

6.3.2.1 Introduction

Analogously to the previous sub-section, this one introduces a modular approach for the dynamic simulation and articulation of the optimisation procedures for those processes presenting decaying performance. Special attention needs to be paid when postulating a validation scheme for such kind of processes since internal disturbances, external actions and changes in the model play a significant role. A proper framework should have in favour some desirable features related the software modularity. This aims to encapsulate the different functionalities and to facilitate the software maintenance and reuse, taking advantage of the plug and play philosophy.

Following, the main ideas about how the simulation/optimisation has been carried out, according to the previous criteria, are summarised. Next section provides roughly the hybrid discrete and continuous processes modelling basis. After that, the way in which external and internal changes are simulated is shortly described. Finally, the approach applied for the simulation-optimisation of the systems is given.

6.3.2.2 Dynamic process simulation

The continuous physical behaviour of a process system is expressed mathematically by sets of nonlinear differential and algebraic equations (DAEs) of the form:

$$f(\frac{ds}{dt}, s, x, t) = 0 \tag{6.10}$$

[4]Global CAPE-Open project, European programme of Brite-EuRam IMS 26691. The CAPE-Open standards are currently published and maintained by the CO-LAN consortium (www.co-lan.org).

where $s(t)$ and $x(t)$ are sets of variables representing the system conditions and the external inputs respectively and $\frac{ds}{dt}$ represents the derivatives of $s(t)$ with respect to time t. Models of the form (equation 6.10) describe most lumped systems. If, in addition, the system properties are also functions of spatial position, that is, distributed systems, the physical behaviour is expressed by mixed sets of partial and ordinary differential and algebraic equations.

6.3.2.3 Hybrid process

Almost all process systems of practical interest are hybrid in nature, exhibiting both discrete and continuous characteristics. Within relatively narrow ranges of operating and environmental conditions, the fundamental laws of physics expressed in the macroscopic scale (which is normally of interest to process engineering) naturally give rise to continuous mathematical descriptions. However, outside these ranges, discontinuous behaviour arises routinely due to phase transitions, flow regime transitions, equipment geometry and a vast variety of other factors.

Discrete/continuous behaviour is an intrinsic characteristic of most process systems even when subject to continuous external inputs. The latter comprise a variety of disturbances, over which one might have no control, and deliberate control actions and manipulations imposed on the plant as a result of process operation and control. In reality, these inputs may themselves be discontinuous and in general may depend on the state of the process itself. Indeed, any process that is operated in an essentially dynamic manner, such as a batch process or a periodic separation process, will experience frequent control actions of a discrete nature in order to maintain operation in a time dependent, often cyclic, mode.

6.3.2.4 Mathematical modelling of hybrid systems

Hybrid discrete-continuous behaviour can be expressed mathematically using the following formalism termed State Transitions Networks (Barton, 1992; Barton and Pantelides, 1994; Dimitriadis et al., 1997). It is assumed that a system may exist in N discrete *states*, $\mathbf{S}_k$ $k = 1 \ldots N$, each described by a potentially different set of variables and/or equations:

$$F_k[s_k, x, t] = 0 \quad \forall k = 1 \ldots N \tag{6.11}$$

where $F_k[\cdot]$ is in general a nonlinear operator corresponding to the functional form of equation 6.10.

In addition to the system states, a set of possible *transitions* $\mathbf{T}_{kk'}$ between pairs of states (k, k') is also defined. A transition is characterised by two major attributes:

- A set of *logical conditions* $L_{kk'}$ which determines when a transition is triggered. These conditions are expressed in terms of the variables in the originating state k and are of the form:
$$L_{kk'}[s_k, x, t] \tag{6.12}$$

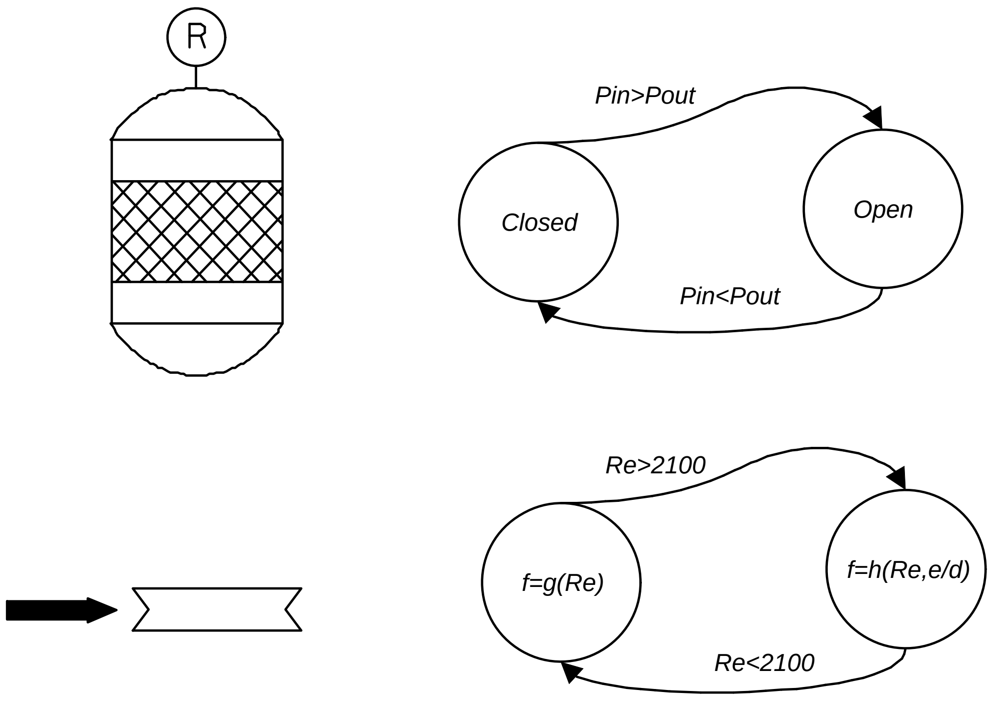


Figure 6.20: Relief valve and flow transition as examples of internal changes

where $L_{kk'}[\cdot]$ is a scalar boolean function that may involve arithmetic comparison operators $(=, \neq, \leq, \geq, <, >)$ as well as the boolean operators (AND, OR, NOT, ...).

- A set of *relations* determining the condition of the system in the destination state k' immediately following the transition. In general, these may involve the variables in both the originating and the destination states:

$$I_{kk'}[s_k, s_{k'}, x, t] = 0 \tag{6.13}$$

It is assumed that 6.13, together with the equations 6.11 describing the system behaviour in state $\mathbf{S}_{k'}$, i.e.:

$$F_{k'}[s_{k'}, x, t] = 0 \tag{6.14}$$

define unique values for $s_{k'}$ at the time of the transition and a unique system trajectory thereafter.

A very simple example of state-transition network that describe an internal change is given by the vapour escape in relief valve of a reactor (figure 6.20). Analogously, a fluid that change of regimen flows from laminar to turbulent constitutes also an internal state change.

Managing Internal discontinuities As previously mentioned, although the simplest mechanism for specifying discontinuous equations are IF/THEN/ELSE statements, it is not a sufficiently general mechanism for managing, in a general way, the complex phenomena occurring in chemical processes (i. e. hysteresis, irreversible discontinuities, etc.). The use of states and transitions offer a powerful tool for such purposes. In that context, any model will contain a set of models (for the states) and a set of transitions conditions. Exactly and only one state will be active at time. On the other hand, the transitions are characterised by a logical condition, which in case of being true will trigger a change of state. Such change of state is termed *event*.

Managing external actions From a mathematical point of view, external actions provide additional conditions that determine the process input variables x. Such conditions may take the form of explicit assignments (e.g. $x = x(t)$), representing, for example, external disturbances or open-loop control actions, or, more generally, they may also involve the process variables s, thereby allowing the description of feedback control mechanisms. In any case, external actions can be viewed as one or more separate subsystems each described by its own state-transition network. These subsystems can then be combined with the subsystems describing the process physics as defined above (equations 6.11 to 6.14) to form a state-transition network of the entire hybrid process system.

6.3.2.5 Current capabilities of commercial simulators

The current state-of-the-art (equation-oriented, as gPROMS and ASPEN CUSTOM MODELER) process modelling and simulation tools utilise task languages for the description and modelling of external control actions, manipulations and process operations. Central to this methodology is the idea that the mathematical modelling of a process system can be decomposed into two distinct parts:

- the physical model describing the physical behaviour of the system over the entire time horizon of interest, allowing the state-transition approach.
- a simulation description comprising a schedule of task entities representing the external actions applied to the system during this period.

This provides a natural approach to process modelling and simulation which emulates the manner in which chemical process plants are operated in reality. A schedule of tasks mirrors the action of an operator or control system on the physical process plant. For example, during the simulation of a start-up procedure the schedule of tasks will manipulate the model entity just as operators manipulate the plant itself by opening valves or placing control loops under automatic control.

For the simulation of logical actions, modular-sequential simulators (e.g. HYSYS) offer tools for easily organising the external tasks in a hierarchical way, analogously to the procedural control. Nevertheless, it is, unfortunately, very difficult to specify internal states in sequential-modular simulators, because the equipment models are encapsulated, acting as black box models. Personalising such models requires programming, rather than configuring. In any case, the fact that they offer tools for organising external tasks allows the simulation of complex external actions, like for instance those required for the detailed simulation of an start-up or shut-down operations. Nevertheless, it appears that modular-sequential simulators still posing many difficulties for a proper simulation of hybrid systems.

6.3.2.6 The simulation/optimisation framework

The systems considered in this thesis work (the filter example, the evaporators and reactors) have associated two states clearly differentiated: the normal operation, with decreasing performance, and the maintenance operation, such as cleaning and regeneration. Thus, and since the internal disturbances are the main protagonist for such problems, equation based simulators are the appropriate choice to suitably represent the systems. Consequently, ASPEN CUSTOM MODELER has been used.

Regarding the external actions, and in correspondence with the states, there are two main tasks: to operate the equipment and to perform the maintenance task. These tasks have associated all the information and actions required to properly operate and clean (for instance) the equipments. Additionally, and what is more important from the point of view of this thesis, the final condition associated to the operation task constitutes the bridge with the proposed methodology.

As it can be foreseen, the problem consists in that although the task hierarchy is trivial, the event that trigger the change from one task to the other requires the continual execution of the RTE procedure (chapter 4) rather than the evaluation of a simple logical condition. The diagram given in the figure 6.21 represents the static architecture used in order to deal with this issue concerning the simulation of the system. Five different elements are identified:

- *The Simulator block*, which includes the models associated to the states (operation and maintenance) for the scenario under consideration.
- *The Tasks block*, which includes the tasks, mirrors of the states, responsible for the executions of the required actions to “operate” the model during the corresponding states. According to the case, the execution of a task involves from complex manipulation to just triggering a change of state in the *Simulator* block.
- *The Database*, which stores the key information generated by the simulator, such as historical values. Besides, economical data is also stored, as well the equations required for evaluating *IOF* at every time stamp.
- *The RTE block.* It implements the procedure explained in chapter 4, section 4.4 using the information available at the *Database*.
- *The GUI*, which allows the interaction with the user.

It should be noted that the substitution of the model and parameters, do not imply any change in the rest of the system. This feature is particularly desirable when several different cases are

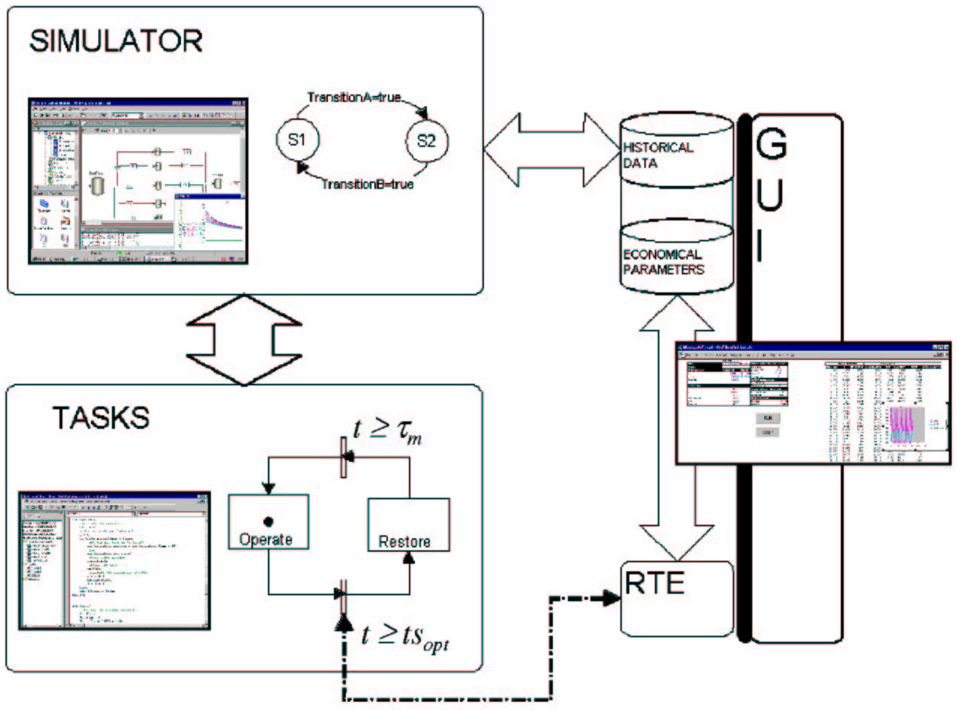


Figure 6.21: Static structure

considered, as is the case in this thesis work. Furthermore, it is worth noting that the industrial implementation corresponds to the replacement of the *Simulator* block by the *Plant*, and the *Tasks* block by the *Procedural Control System*. There is, however, a subtle complication regarding the dynamic structure used for simulation purposes since certain control over the simulator integrator is required. This fact can be seen in the figure 6.22, which shows a sequence diagram corresponding to the simulation of plan with a given number of cycles the single equipment case.

When required by the user, a main program establishes the connection with the simulator. Data form the *Database* are then cleared and the simulation reset. Then, the main program invokes sequentially, for every cycle, the operation and maintenance *Tasks*. The operation task continuously sends the data obtained from the *Simulator* to the *Database* and asks RTE to perform the required calculations in order to evaluate its final condition. On the other hand, the maintenance task acts analogously to the operation one, and contrarily to the latter, has a simpler final condition (the elapsed time required for perform the maintenance task). After each cycle is finalised, a summary is generated and stored in the *Database*. When all the cycles have been simulated, the main program closes the connection with the *Simulator* block. The GUI (Graphical User Interface) continuously display the *Database* information during the simulation, and allows to the user to start, pause and stop the run, as well to specify and change any parameter. When several equipments are in parallel, the scheme still being the same, with the added complexity of

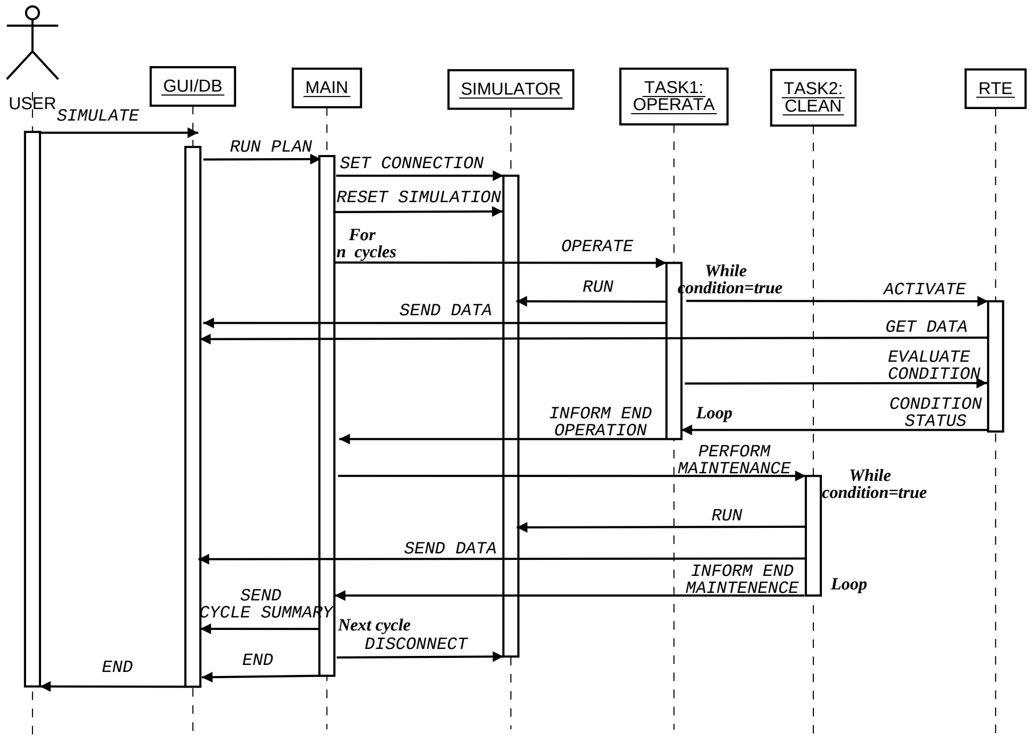


Figure 6.22: Dynamic structure

having more models and tasks (two for every equipment respectively).

6.3.2.7 Final comments

A simple framework for the simulation and optimisation of process with decaying performance has been introduced. Again, the proposed architecture have deal with the potential problems by taking assigning properly the functionalities to the available tools. The modularity allows to focus the efforts on the system communication and the decision-making, but not in the development of its constitutive parts, in addition to the use of plug and play philosophy. The benefits of the proposed structure has been manifested when being applied over the scenarios early presented in chapters 4 and 5 (e.g. the filter, furnaces and evaporators scenarios).

6.4 General Conclusions

This chapter focussed on some complementary but not less important issues at the time of implementing on-line optimisation systems.

On the one hand it has been given insight to the influence of RTE parameters on the process economical performance. Besides, a methodology for an adequate tuning of such parameters has been also proposed. It is shown how the parameters related to the decision variables changes can be tuned just by using the steady state model and information about the variability of the disturbances. On the other hand, the RTE period parameter needs both, the characterisation of the disturbance in terms of amplitude and frequency and a further testing over a dynamic simulation of the process. It has been also shown how the controllers affect the economical performance, and that even though, the perfect control assumption is a quite good reference since properly represents the trend. In addition, it is indicated that on-line optimisation layer does not affect substantially the system stability, but that special care should be taken with the steady state detector specification and highly non-linear systems.

On the other hand, and as explained in section 1.4.2 (page 12), on-line optimisation systems may fail because inadequate software reliability issues. Thus, software frameworks for the development and implementation of the essential elements of an on-line optimisation system has been also considered. The advantages of the proposed architectures are that known tools for the chemical engineer are the main elements used and the modular approaches which allows the use of plug and play philosophy. The benefits of the proposed structure has been manifested when being applied over the scenarios early presented in chapters 3, 4 and 5.