EngOpt 2008 - International Conference on Engineering Optimization
Rio de Janeiro, Brazil, 01 - 05 June 2008.

A New Tool Providing an Integrated Framework for Process Optimization

E.C. do Valle; R.P. Soares; T.F. Finkler; A.R. Secchi

Group of Integration, Modeling, Simulation, Control and Optimization of Processes GIMSCOP (Department of Chemical Engineering, Federal University of Rio Grande do Sul - UFRGS) Rua Luiz Englert, s/n CEP: 90040-040 - Porto Alegre - RS - BRAZIL E-mail: arge@enq.ufrgs.br

1. Abstract

Nowadays, process optimization plays an important role in the chemical industries, providing many benefits. The aim of this work is to present a new framework for process optimization of chemical industries based on rigorous process models and with many integrated features, such as: customized thermodynamic data package, dynamic and steady-state simulation, optimization, parameter estimation, data reconciliation, gross error detection and integration with data bank. The integrated framework with all these features increases significantly the benefits of process optimization. This tool, named EMSO (Environment for Modeling, Simulation, and Optimization), is an equation-oriented dynamic and steady-state simulator and optimizer. It has an equipment library with rigorous models that can be viewed and edited by the user, using a high-level modeling language. The process flow diagram can be built either in text or graphical mode and the equipments can be modeled using an object-oriented approach. This software provides a set of sparse algebra and automatic differentiation feature to solve NLA (Non-Linear Algebraic Equations) systems in a fast and efficient way. NLP solvers, such as IPOPT and OPT++, were interfaced with this framework taking advantage of EMSO's efficiency to solve NLA systems, reducing the computational effort required to achieve the solution of optimization problems. Other optimization solvers can be readily interfaced through a plug-in system. Besides steady-state optimization of process flow diagrams with rigorous models it is also possible to perform steady-state and dynamic parameter estimation using the optimization feature. Furthermore, given a set of process measurements, it is also possible to solve data reconciliation problems and perform gross error detection. A language to describe the formulation of the optimization problems was developed and is presented in detail. The implementations mentioned above were evaluated with literature and practical examples, showing the benefits of using the proposed framework.

2. Keywords: Chemical Process, NLP Optimization, Parameter Estimation, Data Reconciliation, EMSO.

3. Introduction

Nowadays, with the increasing development of the personal computers capacity, chemical processes simulators has become more popular among academy and industry. Although process simulators can help the process engineer in many tasks, such as:

- Process evaluation at different operating points;
- Plant start-up and shutdown;
- Process scale-up;
- Grades transition;
- Advanced process control;

they lack of some functions that process engineers may need, such as:

- Perform parameter estimation (kinetic constants, efficiency, thermodynamic and others) based on process data fed into the simulator;
- Perform process data reconciliation (such as: flowrates, composition, temperatures);
- Perform steady-state optimization that can determine the optimal operating condition to reach some desired goals;

- Perform process synthesis (such as: heat exchanger, mass exchange networks, reaction/separation synthesis, distillation columns sequencing and others) based on optimization methods that can handle continuous and discrete variables.
- Perform dynamic optimization which, in the chemical processes case, can help to determine which of the process variables must be modified and in which sequence in time, to reach a desired goal

In this context, the aim of this work is to present a new framework for process optimization of chemical industries based on rigorous process models with many integrated features. This tool, named EMSO, Environment for Modeling, Simulation and Optimization [1] was initially designed as a generic steady-state and dynamic simulator for chemical and petrochemical process modeling. The simulator is object oriented and the code describing equipments and units can be reused between flowsheets, simplifying the modeling process. For example: a model of a mixer that considers the energy balance of the mixture can be modeled reusing a model of a mixer that considers only mass and components balance with the energy balance added. Since EMSO has a dynamic simulator feature, it is possible to monitor the process variables along the time, allowing the simulation of plant start-up and shutdown and also operating-point transition. The software has many features such as its own graphical interface which can plot graphical results, characterization of differential-algebraic system of equations , the use of sparse algebra algorithms, automatic differentiation, detection and solving high-index differential-algebraic equation system, an unit consistency checker, a plug-in system which allows the use of external routines (such as solvers and thermodynamic packages) and others. Nowadays, EMSO has a vast rigorous model library, which includes the main equipments used in the chemical and petrochemical industry such as separators (flash, distillation columns), heat exchangers, reactors, pumps, valves, controllers, and many others. The models presented in the library are opened and can be viewed edited and added by the users. Considering that EMSO can be interfaced with external calculation tools, the software was integrated with external optimization solvers, in order to increase its functionalities, as described in the next section.

4. Methods

Before state and define how the simulator would handle the parameter estimation and data reconciliation problems it was necessary to perform the communication between the optimization solver and EMSO. These tasks were performed in three stages:

- Development of the optimization language
- Choosing the optimization solver
- Development and implementation of a communication interface between the optimization solver and the simulator

It was defined, also, that the optimization problem could be written in two ways: in the first one, the user defines and writes all the variables and equations involved in the optimization problem, whereas, in the second, the user imports the variables and equations of an already defined simulation/flowsheet (at this moment, only steady-state flowsheets can be imported). In this last case, it is necessary to free some variables in order to have the necessary degrees or freedom to run the optimization.

Having the optimization framework running, other related features were implemented in order to increment the software functionalities, such as parameter estimation using algebraic and differential system of equations and data reconciliation.

4.1 Optimization Language

The optimization problem starts with the keyword Optimization and ends with the word end. The problem description is divided in five subsections:

- VARIABLES: where the decision (or free) variables that integrate the problem are declared. Here it is also defined the upper and lower bounds and also the initial guesses for the variables.

- MINIMIZE or MAXIMIZE: where the objective function is written. It is possible to perform a maximization or minimization of multi-objective functions.
- EQUATIONS: where the equality and inequality constraints are written.
- OPTIONS: where the solver specific options are set.
- FREE: in the case where the equations of a flowsheet are used (such as mass and energy balances) it is possible define here which of the variables are allowed to be free by the optimization. In the case where all equations are written in the EQUATIONS section, and all the variables are declared in the VARIABLES section, it is not necessary to use this section.

In order to present the language, the optimization test problem proposed by [2], listed in Eq. (1) is listed below.

$$\begin{cases} min_{x_1,x_2,x_3,x_4} & x_1.x_4.\left(x_1+x_2+x_3\right)+x_3 \\ \text{subject to} & \\ 1 < x_i < 5 & i = 1..4 \\ x_1.x_2.x_3.x_4 \geq 25 & \\ x_1^2+x_2^2+x_3^2+x_4^2 = 40 & \end{cases} \tag{1}$$

The problem above, written in the proposed language have the following form:

```
Optimization hs71
 VARIABLES
   x1 as Real(Default=1, Lower=1, Upper=5);
   x2 as Real(Default=5, Lower=1, Upper=5);
   x3 as Real(Default=5, Lower=1, Upper=5);
   x4 as Real(Default=1, Lower=1, Upper=5);
 MINIMIZE
   x1*x4*(x1+x2+x3) + x3;
 EQUATIONS
   x1*x2*x3*x4 > 25;
   x1*x1 + x2*x2 + x3*x3 + x4*x4 = 40;
 OPTIONS
   NLPSolver = "opt_emso";
   NLPSolveNLA = 1;
   relativeAccuracy = 1e-6;
end
```

In the case where previously built flowsheets are used, as in the optimization ”FlashOpt1” below, there is a previously defined flowsheet, ”FlashSteadyTest”, which is imported in the first line statement. In this optimization, the objective is to maximize the lightweight compounds at the flash's vapour stream, the problem has the following form:

```
Optimization FlashOpt1 as FlashSteadyTest
  MAXIMIZE
    light;
  FREE
    fl.OutletL.T;
  EQUATIONS
    fl.OutletL.T < 320 * 'K';
    fl.OutletL.T > 300 * 'K';
  OPTIONS
    Dynamic = false;
    NLPSolveNLA = false;
    NLPSolver( File = "ipopt_emso", RelativeAccuracy = 1e-6);
end
```

4.2 The Choosing of the Optimization Solver

The use of object-oriented software libraries for optimization is a future tendency, as presented in [3], since one of the goals of object-oriented software is to allow plug and play features in the formulation of the optimization models. In this context, two solvers were evaluated: OPT++ [4] and IPOPT [5], [6]. OPT++ is a free software (LGPL license), written in C++, developed by Sandia National Labs. IPOPT is developed as part of the COIN-OR project sponsored by IBM, it is also free (CPL license, less restricted when it comes to distribution) and also written in C++. Both are routines that handle non-linear equality and inequality constraints (very common in Chemical Engineering problems), and use interior point algorithms (see [7]). The main advantage of IPOPT over OPT++ is that the former is specially designed to deal with large-scale problem, also common in Chemical Engineering problems when an equipment model is written in a rigorous way. As the simulator EMSO, IPOPT has special sparse algebra routines that reduce the memory used to allocate the Jacobian matrix, so this data is passed from the simulator to the optimization engine in a more efficient way.

4.3 Communication Interface between the Simulator and the Optimization Solver

The communication interface was made to use the automatic differentiation (for the objective function and constraints gradients) and the sparse algebra routines of EMSO. The optimization problem solutions are obtained by the following steps:

1. The user formulates the optimization problem and selects the optimization solver (OS) in the EMSO framework.
2. Problem initialization by the optimization solver.
3. An automatic differentiation of the objective function and constraints is performed by EMSO and the structure of the gradients/Jacobian matrix is passed to the OS.
4. The OS requests the objective function and the constraints (equality and inequality) residuals and gradients (Jacobian) to EMSO.
5. EMSO calculates the objective function and the constraints residuals and gradients and passes the values to the OS as sparse (compacted) matrix.
6. A new step or search direction is calculated by the OS. Steps 2 to 4 are repeated until convergence or non-convergence condition is met.
7. The OS returns the results to EMSO.
8. Results are published by EMSO.

In the case where a flowsheet optimization is executed, it is possible to solve the system of equations (resulting from the flowsheet) considering only the equality constraints (as a non-linear-algebraic system of equations, or NLA) before the optimization, in order to present a feasible initial guess for the OS. It is important to notice that in the case where thermodynamic properties are needed, EMSO make a call to the thermodynamic property package that retrieves (from a data bank) or calculates the desired property, which may take extra computing time. Since the current version of EMSO still does not provide the Hessian matrix by automatic differentiation, the OS must calculate it by finite differences (or BFGS, depending on the OS option available) and, therefore, it is possible to have loss in the Hessian matrix accuracy and an extra computational time is required comparing with a case where this matrix is already provided.

4.4 Parameter Estimation problem definition

Initially, the feature of parameter estimation was developed to deal with two types of estimation problems: the NLA (or steady-state) estimation and the DAE (or dynamic) estimation. In this problem formulation, differently from the pure optimization, the equations involved in the estimation problem must be previously defined in the FLOWSHEET section. The problem description is divided in three subsections:

- ESTIMATE: where the parameters that will be estimated, previously declared in the flowsheet section, are selected and set-up. The parameters initial guesses and their limits can be overridden in this section;
- EXPERIMENTS: where the data file with the measured results is set-up;
- OPTIONS: where the solver and estimation specific options are set.

The description of an estimation problem is described below:

```
Estimation Biop_NE_Estt5 as Biop_NE_process5t
  ESTIMATE
    #PARAMETER  START     LOWER  UPPER  UNIT
    Kss         0.2009    0.004  7      'kg/m^3';
    Ksn         0.0446    0.005  1      'kg/m^3';
    mim         0.7979    0.1    0.8     '1/s';
    alf         2.0293    1      5         -  ;
    gam         0.0850    0.05   5      '1/s';
    K1          0.4059    0.1    3      'kg/kg';
    K2          -0.00795  -1     3      '1/s';
    Yn          10.62     0.1    18        - ;
    kd          0.007     5E-2   1      '1/s';
  EXPERIMENTS
    # DATA FILE    WEIGHT
    "Bio.dat"      1;
  OPTIONS
    Statistics( Fits=true,Parameters=false,Predictions=false);
    NLPSolver(MaxIterations = 1000,File = "complex");
end
```

The flowsheet candidate to estimation, can be a simple laboratory equipment or a complex industrial plant. In the last case, the estimation procedure may require a huge computational effort, since many equations and variables are involved. In order to reduce the computation effort required by the computer and increase the problem solving robustness, a two-level methodology was proposed in the estimation problem, based in the fact that the objective function (the sum of square errors) of the parameter estimation problem involves just a few parameters. In the two-level technique proposed, the problem is divided in two parts: the first one involves just the equations and parameters directly present in the parameter estimation problem, these data will be "viewed" by the optimization solver only, while the other problem will be handled in the simulation engine. These procedure is better explained in the next section.

4.4.1 Parameter Estimation separation methodology

In the normal simulation a system of non-linear algebraic (NLA) or dynamic non-linear algebraic equations (DAE) is solved based in a set of specifications (or specifications and initial conditions, in the case of DAE systems):

$$F(X) = 0 \tag{2}$$

where F is the vector of the equations and X the vector the the variables. In the case where we have an estimation problem, we have the following formulation:

$$\begin{cases} min_\alpha & S(\alpha) \\ \text{subject to} & \\ F(X(\alpha)) = 0 & \\ -\infty \leq \alpha \leq \infty & \end{cases} \tag{3}$$

where the S is the objective function (sum of square between the model and the observed data), X is the variables of the model F, α is the decision variables (the model parameters to be estimated).

Since F can have hundreds or, depending on the case, thousand of equations and variables, considering X also as decision variables can make the optimization problem with huge dimensions, that may require a great computational effort, specially when the the optimization solver does not support sparse algebra. So the problem is separated between the decision variables of the optimization problem, (the parameters α) and other variables and equations only involved in the NLA or DAE system. The algorithm describing the problem solving is presented below:

1. The simulator splits the equations according to the procedure above.
2. The optimization is started and the initial values of α and X are passed to the simulation engine.
3. The engine set these values as specifications, solves the NLA or DAE and send these values to the optimization solver (OS).
4. The OS takes these values, calculates a new search direction and step and send them again to the simulation engine.
5. The steps 3 to 4 are repeated until convergence is acquired.

4.5 Data reconciliation problem definition

The parameter estimation and data reconciliation are very similar problems: both are optimization problems with an objective function and equality and inequality constraints. For the parameter estimation, the objective is to find the parameters that minimize the squared error function between the model data and the observed data; for the reconciliation, the objective is to find the variables values that minimizes the squared error function between the own variable and its measures. As previously made for the estimation problem, first a reconciliation language was developed and presented below

- RECONCILE: where the variables of the flowsheet to be reconciled are selected.
- FREE: where the variables specified (fixed) in the flowsheet simulation are freed to allow the optimization to be executed.
- OPTIONS: where the solver, reconciliation and error detection specific options are set.

The two-level procedure performed to the estimation is also applied to the reconciliation case. The description of an estimation problem is described below:

```
Reconciliation HeatEx_Rec as HeatEx_Flow
  RECONCILE
    x1; x2; x3; x4; x5; x6;
  FREE
    x1; x2;
  EXPERIMENTS
    ## FILE           WEIGHT
    "heatEx.dat"        1;
  OPTIONS
    Filter = "mean";
    Significance = 0.95;
    GrossErrorTests(Global = true, Nodal = true, Measurements = true);
    NLPSolver(MaxIterations=1000, File = "complex");
    Dynamic = false;
end
```

5. Results

In order to test the optimization results, four problems from literature were evaluated, and its results are presented in the following sections.

5.1 Flowsheet optimization

The first example used is the Ammonia production process presented in [8] using rigorous models. For the thermodynamic calculation, the Assimetric Peng-Robinson equation of state for the liquid and vapour phase was used. The thermodynamic property package used was VRTherm [10]. The problem has 219 variables and 203 equations and it is presented in Figure 1. The objective of this optimization is to minimize the heat load in both flash units after the ammonia reactor. The decision (or free) variable selected to attain this goal is the split fraction at the top of the flash unit after the reactor, that splits the vapour flowrate between the compressor (recycle) and the purge. It was also added some extra constraints to assert some economic conditions, such as:

1. The ammonia loose, defined as the molar flow of ammonia in the purge must be less than 1 lbmol/h;
2. The production of ammonia, defined as the molar flow of the ammonia in the botton stream of the second flash separator must be greater than 90 lbmol/h, or
3. The production of ammonia, defined as the molar flow of the ammonia in the liquid stream of the second flash separator must be greater than 101 lbmol/h.

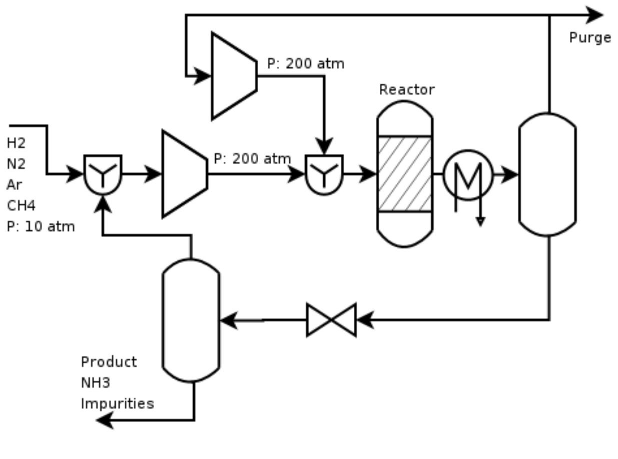

Figure 1: Ammonia production plant presented in [8].

In order to compare the results, two optimization were performed. One considering the constraints 1 and 2 and other considering constraints 1 and 3. As can be observed, the constraint 3 is more ”tightened” since it imposes a greater production rate of ammonia. Both optimization were compared with the case where only simulation was considered. These results are presented in Table 1. When compared with the case where only a simulation was performed, the optimization reduced the heat load, respecting all constraints.

Table 1: Ammonia plant result.

Case	Total heat load(KW)	Split fraction	Purity	Production (lbmol.h-1)	Loose (lbmol.h-1)
Simulation only	-3372.95	0.78	0.998	97.99	1.04
Optimization with production > 90 lbmol.h-1	-3368.77	0.89	0.998	99.79	0.5
Optimization with production > 101 lbmol.h-1	-3364.75	0.999	0.998	101.49	3.3E-2

5.2 Parameter Estimation

In order to demonstrate the parameter estimation feature in EMSO, two estimation cases are presented: one considering a simple steady state estimation, and other, a dynamic parameter estimation.

5.2.1 Antoine's equation parameter estimation for naphthalene

In this example the parameters A, B and C of the Antoine's equation for naphthalene are estimated based on the P_V vs. T data obtained from [9]:

$$ln(P_V) = A - \frac{B}{T + C} \tag{4}$$

The results of the parameters were the same as described by the author and is presented in Table 2.

Table 2: Antoine's equation parameter estimation.

Parameter	Value	Unit	Confidence Interval
A	6.2	-	0.392
B	2013.6	K	192.88
C	171.5	K	12.02

5.2.2 Biopolymer Bioreaction Parameter Estimation

In this example, a biopolymer (poli 3-hidroxibutirate) production reaction was considered and modeled as a system with 7 equations, 3 differential and 4 algebraic with 9 parameters [11]. A total of 22 experimental points were used in the parameter estimation. The results of the parameter estimation of this problem is presented in Figure 2 where the biomass and biopolymer production from experimental data points are plotted with the predicted results from the parameter estimation of the dynamic model. In order to see more clearly the results, a small time-window of the whole experiment was zoomed in this plot.

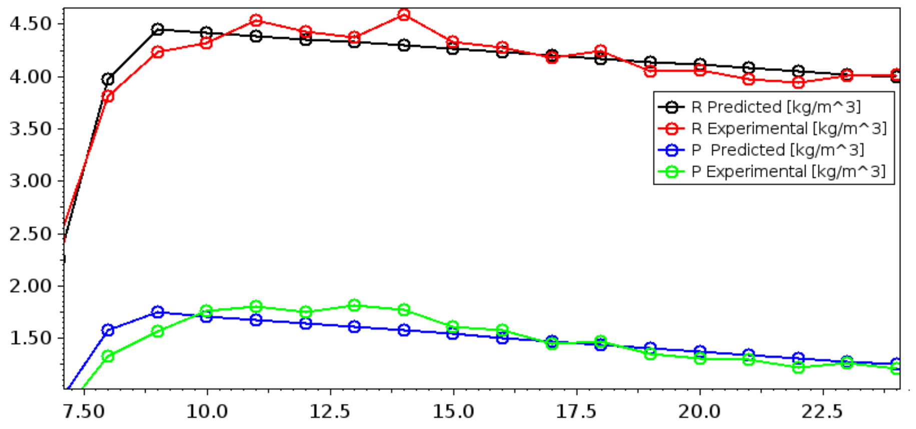

Figure 2: Biopolymer example - experimental vs. predicted data.

5.3 Data Reconciliation

In order to test the EMSO's data reconciliation feature, a classical example of data reconciliation was considered [12]. In this example, a simple system of a heater with a by-pass valve with all streams measured is evaluated as presented in Figure 3. The results are presented in Table 3 and are the same obtained by the author, considering the numerical precision.

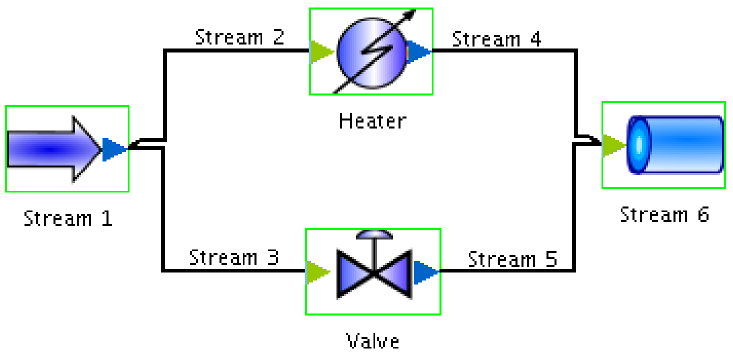

Figure 3: Heater with by-pass from [12].

Table 3: Reconciliation problem results.

Stream	Measured	Reconciled
1	101.91	100.25
2	64.45	64.5
3	34.65	35.74
4	64.20	64.5
5	36.44	35.74
6	98.88	100.25

6. Conclusions

In this paper, an optimization tool based on a simulation engine is presented. The process simulator, initially designed to perform steady-state and dynamic simulations had its features increased when the optimization was integrated with the software, such as: simple optimization and flowsheet optimization, parameter estimation with algebraic and differential models and data reconciliation. The software can optimize from simple problems (as the Hock and Schittkowski problem 71) to complex chemical units (the ammonia plant, presented in [8]). The object oriented approach used to model the equipments and units can lead to high reuse of the already built library model, reducing the modeling time necessary to describe complex industrial units. Other advantage is that EMSO's model library is free and opened, as a consequence, the user has the possibility to customize them, differently from the most common commercial process simulators.

7. References

[1] Soares R P and Secchi A R. EMSO: A New Environment for Modeling, Simulation and Optimization. In: *Proceedings of the 13th European Symposium on Computer Aided Process Engineering (ESCAPE 13)*, Lappeenranta, Finland, 2003, 947-952.

[2] Hock W and Schittkowski K. Test Examples for Nonlinear Programming Codes. Lect. *Notes in Economics and Mathematical Systems*, 1981, 187.

[3] Grossmann I and Biegler L. Future Perspectives on optimization - Part II. *Comp. and Chem. Eng.*, 2004, 28, 1193.

[4] Meza J C, Oliva R A, Hough P D and Williams P J. OPT++: An Object-Oriented Class Library for Nonlinear Optimization, *ACM Transactions on Mathematical Software*, 2007, 33, 12.

[5] Wächter A. An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering, Phd Thesis, Carnegie Mellon University, 2002.

[6] Wächter A and Biegler L T. On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming, *Mathematical Programming*, 2006, 106, 25-57.

[7] Rao S S. Engineering optimization : theory and practice. *3rd Edition. New York: John Wiley*, 1996, .

[8] Biegler L T, Grossmann I E and Westerberg A W. Systematic Methods of Chemical Process Design. *Englewood Cliffs: Prentice Hal*, 1997.

[9] Stephenson RM and Malanowski S. Handbook of Thermodynamics of Organic Compounds, *New York: Elsevier*, 1987.

[10] VRTech Tecnologias Industrias Ltda. VRTherm. Porto Alegre, Brazil, 2005.

[11] D.J. Luvizetto. "Cultivo da bactria Bacillus megaterium para a produo do biopolmero poli(3-hidroxibutirato) e modelagem matemtica do bioprocesso". M.Sc. Thesis, PPGEQ/UFRGS, Porto Alegre, RS, 2007.

[12] Narashinham S and Jordache C. Data Reconciliation and Gross Error Detection - An Intelligent Use of Process Data. *Houston: Gulf Publishing Company*, 2000.