Modelling of Combined Cycle Power Plant Ageing in Matlab and Industrial^{IT}®

A. Stothert, E. Gallestey and G.E. Hovland

Automation and Control Systems Group,
ABB Corporate Research,
CH-5405 Baden-Dättwil, Switzerland,
Fax: +41 56 486 7365

Abstract

This paper describes the development of a prototype decision support product that indicates to a power plant operator the effect of daily operation on plant lifetime consumption and recommends short-term operating strategies that optimise plant economic performance. The paper focuses on the use of Matlab/ Simulink models, data structures, and user interfaces used in this decision support tool. The complexity of the modelling requirements and sheer number of parameters resulted in extensive use of Matlab structures and custom made Simulink libraries and proved to be the only viable method of creating a useable system. The final solution is integrated with ABB's Industrial^{IT}® architecture.

1 Introduction

Power plant owners operating in deregulated energy markets are under increasing pressure to improve their maintenance strategies, minimise plant operating costs, and manage the market value of their physical assets. A central component in achieving these requirements is the accurate calculation of production costs — plant depreciation and degradation resulting from production must be included in this calculation. In order to support power plant owners in this competitive deregulated environment a tool recommending plant operating strategies has been developed. Recommending such power plant operating strategies is based on the optimisation of an objective function that includes terms for revenues from energy sales, production costs and plant ageing.

The developed tool focuses on combined cycle power plants which typically includes a gas turbine, heat recovery steam generator and steam turbine. The gas turbine exhaust gas is used to heat a water steam cycle and the resulting steam is used to drive a steam turbine. While topologies vary a typical configuration is for the gas and steam turbine to share a common rotor that is connected to an electrical generator. It is also often possible to operate the steam turbine in a by-pass mode, i.e., the generated steam is sold to process industries or district heating rather than used for generation of electricity. Combined cycle operating strategies are normally chosen by forecasting future energy prices (both electricity and steam) and using "cost of production curves" to select the level of electricity and steam production [4, 5]. In this article the cost of plant ageing is introduced into the cost of production curves. In particular plant ageing is based on models that are directly load dependent and incorporate a memory aspect. This implies that optimisation results in a trade-off between maximisation of immediate profits (i.e., earnings achieved by selling heat and power) and minimisation of lifetime consumption.

Calculating plant ageing is decomposed into calculation of the ageing of critical components within the power plant, for example the steam turbine rotor is one component. Assessing the ageing of each component follows the scheme described in figure 1. The first step is to compute (or measure) all process data applied to the component, e.g., rotor angular frequency. The process data is used to compute the mechanical and thermal stresses applied to the component. The stress profiles are then analysed to identify cycle and hold periods which are in turn used to distinguish between fatigue and creep ageing respectively [1]. The final stage is to apply damage models that compute the fatigue and creep damage.

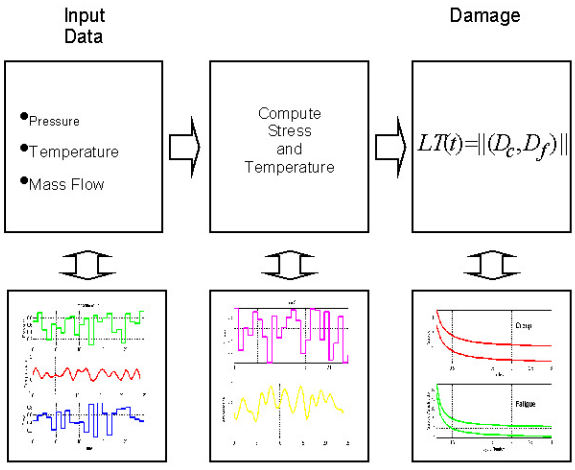

Figure 1: *Stages in the lifetime calculation.*

In the context of an operational decision support tool the next stage is to describe the combined cycle power plant as dynamic system with states relating not only to the plant process variables but also to the component damage states. This is possible by combining process models with the dam-

age models described above. The operational optimisation problem can then be described in a model predictive control framework [9] where the objective function is a weighted sum of the process and damage state variables.

The remainder of this paper is arranged as follows. Section 2 describes the mathematics behind the process, ageing, and optimisation models. Section 3 gives a detailed description of the data structures, user interfaces and flow controls used in the developed prototype product. Before concluding remarks are given in section 5 section 4 briefly describes the strategy for converting the prototype system into a final product based on the ABB Industrial$^{\text{IT}}$®platform.

2 Mathematical Models

This section describes the models used in each of the stages in figure 1. With the exception of the models used to specify the model predictive control problem (see section 2.3) and the stress computations, which are modelled in matlab script files, all the models are implemented in simulink. Specifically both generic combined cycle power plant process models and damage models were implemented as simulink libraries.

2.1 Process models

Space does not permit a detailed description of all the implemented process models. Note that generic models for, pipes, evaporators, heaters, separators, drums and economisers, attemporators, coolers and, turbine stages are sufficient for damage modelling of key components in a combined cycle power plant.

By way of a process example the model of a steam turbine consists of inlet pipeing, a sequence of turbine stages and outlet pipeing. Each stage in the steam turbine is modelled as follows [11]:

The rotor dynamics are described by a first order differential equation relating angular acceleration to applied torques (Power produced minus demand and friction)

$$I\frac{d\omega}{dt} = P - D - c\omega^2$$

The stages are divided into two sections, before and after a vane. Each section is assumed polytropic and modelled as a pipe, the sections (and turbine stages) are connected through mass balance equations, i.e., for stage i and the section before the vane (subscript zero)

$$\begin{aligned} \text{vol}^i\frac{d\rho^i}{dt} &= \text{flow}^i_{\text{in}} - \text{flow}^i \\ \text{where} & \\ \text{flow} &= c^i\rho^i p^i\sqrt{1-\left(\frac{p_0^i}{p_1^i}\right)^{\frac{n+1}{n}}} \\ \text{flow}^i_{in} &= \text{flow}^{i-1}_{out} \\ p_0 &= \text{const}\left(\rho_0^i\right)^n \text{ polytropic flow} \end{aligned}$$

The power generated by each stage is an algebraic function of the thermodynamic state of the stage and the stage geometry.

2.2 Lifetime or damage models

Lifetime assessment of power plant components requires a multidisciplinary approach involving knowledge of design, material behaviour and non-destructive evaluation techniques. The literature available is vast and it is beyond the scope of this paper to give a detailed account of these methods.

As indicated in the introduction two factors are critical for the modelling of damage for optimisation purposes as required here. Firstly the models have to provide a direct relationship between plant load and plant ageing, and secondly the models must capture the operating history of the component. Fracture mechanics provides a framework that satisfies these requirements, in particular the microcrack propagation (MP) approach [2, 3], provides the required model features and gives a good theoretical basis for the calculation of the lifetime of a component.

The main idea is as simple as it is appealing: a microcrack in a critical location or component is *assumed to exist*, and its propagation is modelled until a predefined critical crack size is attained. The growth of the crack is modelled using the machinery developed in fracture mechanics.

While similar crack propagation under fatigue and creep conditions are described by different equations. The generic form is given by (1)

$$\begin{aligned} \frac{d\,a}{dN} &= \frac{C\left(\max(K_{\max}-K_{\min}-K_{\text{th}},0)\right)^n}{K_{\text{crit}}/K_{\max}-1} \qquad (1) \\ \text{where} & \\ K &= \sqrt{\pi a}\sigma F(a/W) \end{aligned}$$

Where a is the crack length, N the number of fatigue cycles, σ the applied stress, W, C, and n component specific constants[1], and $F(a/W)$ a function dependent on the geometry of the component. K is the so called stress intensity factor and the subscripts refer to the maximum and minimum cycle stresses, the threshold stress below which no crack growth occurs, and the critical stress which causes instantaneous destruction of the component. Creep crack propagation is described by an equation similar to (1) but $\frac{da}{dN}$ is replaced by $\frac{da}{dt}$.

In the simulink implementation $F(a/W)$ is implemented as a lookup table, (functions for different geometries are published in handbooks such as [8]) and the creep and fatigue equations are combined into one model. This is possible by using a cycle identification algorithm for the fatigue equation and the simulink reset integrator, see figure 2.

In order to convert the crack model into a cost that can be used in an optimisation problem a last step is required, viz., the crack length is normalised according to a critical value and component replacement cost, equation (2) illustrates

[1] C and n are material dependent and often vary with temperature.

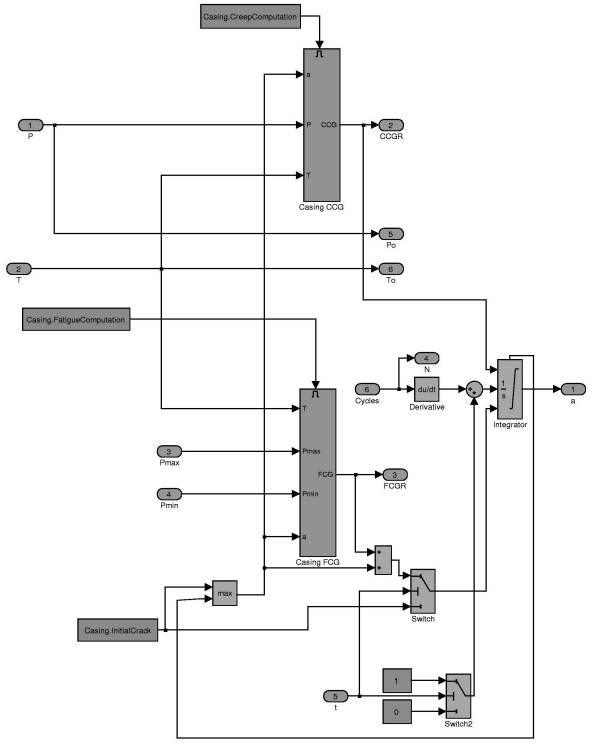

Figure 2: *Combined fatigue and creep crack model.*

this process.

$$C_{\text{damage}} = C_{\text{replace}} \left(1 - \frac{a_{\text{crit}} - a}{a_{\text{crit}} - a_{\text{init}}}\right) \tag{2}$$

2.3 Optimisation models

In order to address the operational optimisation described in the introduction the power plant process and ageing models are represented in a hybrid systems [10] modelling framework known as mixed logic and dynamic models (MLD). This enables the operational problem to be described as a control problem that is solved using model predictive control techniques. In essence the control problem reduces to optimally selecting the on and off times for each of the main plant components and the operating level of each component when it is on. In order to evaluate a particular operating strategy the following are necessary,

- Process models to describe how fuel is converted into electrical and thermal energy. This gives the traditional plant operating costs.
- Process and stress models to indicate component stress loading which in turn is used to compute component damage. This gives the lifetime cost.
- Constraints on the operation of the power plant described in terms of logical formulae, e.g., the steam turbine can not be operated unless the gas turbine is operating

All of the above are available from the models described earlier and are combined into one MLD system where the states represent process variables and component damage. As the MLD framework requires discrete time piecewise linear systems the models in sections 2.1 and 2.2 are discretised via Euler method with the help of the Matlab `feval` function. This process results in a system described by,

$$\begin{aligned} x(t+\Delta) &= Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) \\ y(t) &= Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) \\ & E_2 \delta(t) + E_3 z(t) \leq E_1 u(t) + E_4 x(t) + E_5 \end{aligned} \tag{3}$$

Where

- $x = [x_c, x_b]$ are the continuous and binary states
- $u = [u_c, u_b]$ are the continuous and binary inputs
- $y = [y_c, y_b]$ are the continuous and binary outputs
- δ and z are auxiliary binary and continuous variables respectively. These variables are typically required to model the operating constraints described above.

Note that nonlinear dynamics which would appear on the right hand side of equation (3) can be modelled by piecewise linear functions.

The final stage is to define the objective function that must be minimised by selection of the control input variables. For this problem the objective function is a costs minus revenue function where the costs are a combination of fuel costs and damage or lifetime costs. Using the notation of (3) the power plant operational problem is written as

$$\max \sum_{t=T}^{T+M-\Delta} (\text{Revenue}(t) - \text{Cost}_{\text{fuel}}(t) - \text{Cost}_{\text{lifetime}}(t)) \tag{4}$$

subject to equation (3) and where, for example,

$$\begin{aligned} \text{Cost}_{\text{fuel}}(t) &= k_{\text{fuel}}(t)\text{fuel(t)} \\ \text{Cost}_{\text{lifetime}}(t) &= k_{\text{lifetime}}(LT(t+\Delta) - LT(t)) \\ \text{Revenue}(t) &= p_{\text{power}} \min(D_{\text{power}}(t), \text{power(t)}) + \\ & \quad p_{\text{steam}} \min(D_{\text{steam}}(t), \text{steam(t)}) \end{aligned}$$

Once the MLD model and objective function is defined the advantage is that the operational problem is finally described by a mixed integer linear programming problem. This problem is not solved in Matlab directly but rather by dedicated software which is called from Matlab.

3 Description of Matlab/Simulink Tool

In this chapter we describe the Matlab/Simulink tool in some detail. Section 3.1 describes the data structure arrays. These structures are the backbone of the tool and they have significantly simplified the interfacing to the graphical user interface described in section 3.2. In section 3.3 the functional structure and flowchart of the tool are described.

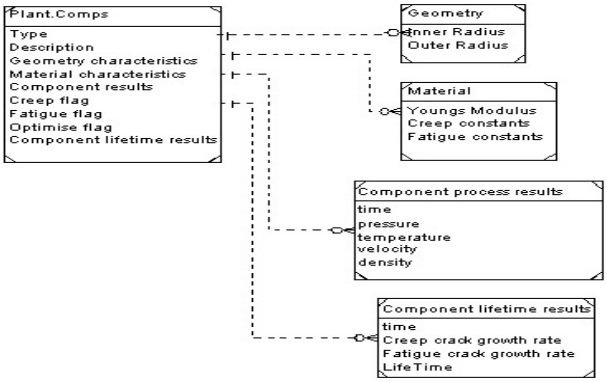

Figure 3: *Power plant component data structure.*

3.1 Data Structures

There is one main plant structure in the tool and for each major aggregate `i`, such as a gas turbine, steam turbine or heat exchanger, there are three main data structures, `Plant(i).Data`, `Plant(i).Comps` and `Plant(i).Results`. The structure `Plant(i).Comps` is illustrated in Fig. 3. The structure `Plant(i).data` contains all configuration data for the particular aggregate component. For example, the data structure for a gas turbine contains values such as maximum load, number of turbine stages, nominal speed, etc. The structure `Plant(i).Comps(j)` contains the smallest components `j` of the plant. A gas turbine, for example, consists of a number of pipes, vanes and blades in addition to the rotor and casing. Each of these smallest components has the data structure shown in Fig. 3. For each component `j` the `Plant(i).Comp(j)` structure contains geometry and material parameters as well as process and ageing simulations. Finally, the structure `Plant(i).Results` contains simulation and optimisation results directly related to the aggregate components, for example produced power of the turbines, rotor speed, inlet flow, etc. All simulation and optimisation results which are not directly related to a component structure `Plant(i).Comp(j)` are stored in `Plant(i).Results`.

3.2 Graphical User Interface

An example of the graphical user interface (GUI) is shown in Fig. 4. The main feature of the GUI is the fact that no hard-coding of presentation data is done. The only information the GUI is given, is the global `Plant` structure with the substructures `Plant(i).Data`, `Plant(i).Comp` and `Plant(i).Results` as described in section 3.1. The GUI loops through all aggregate components of the `Plant` structure to display configuration data, process simulation, ageing simulations and optimisation results. By building the GUI in this way, we have a very clear interface between simulation code and the GUI. Hence, a migration of the simulation code to the ABB Industrial^{IT}® architecture has been significantly simplified. (For further comments about ABB's Industrial^{IT}® architecture, see section 4.)

Moreover, when a completely new power plant is studied, the GUI remains unchanged. The addition of new aggregates, such as gas or steam turbines or a new topology of the heat exchanger (for example single-pressure drum instead of triple-pressure drums) is straightforward from the GUI point of view. For example, the Matlab code for displaying a list in the upper left part of Fig. 4 is given below.

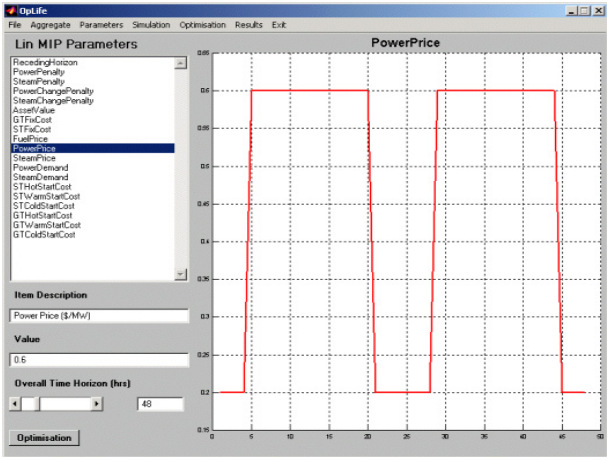

Figure 4: *Graphical User Interface which loops through the data structures and presents physical parameters, process and ageing simulations, and optimisation results.*

```
set(Handles(LIndex),'Value',CurrentAggregate);}
set(Handles(LIndex),'String',{Plant(:).type});}
```

The call `Plant(:)` loops through and displays all aggregate components of the power plant.

3.3 Functional Structure and Flowchart

The functional structure is shown in Fig. 5. The tool consists of a process simulation part, an ageing simulation part and an economic optimisation part. Fig. 6 illustrates the execution sequence from input to output data. Matlab/Simulink provides a powerful solution for modelling combined cycle power plants of large complexity. The example presented here is taken from an operating combined cycle power plant where over 400 equations (process models, crack models, optimisaton functions, etc.) and over 3000 parameters and results are used to describe the plant.

4 Integration with Industrial^{IT}®

Industrial^{IT}® (IIT) is an architecture for seamless linking of multiple applications and systems in real-time. This could include e.g. process automation, asset optimisation and collaborative business processes. It supports application reuse for higher quality and lower engineering costs, and simpler operation, maintenance and training. It includes functionality ranging from field devices to business systems, focused on supporting decisions and improving customer productivity and asset utilization, from the first phases of design, through installation, commissioning, operation, maintenance and asset optimization.

A key tool for migrating the Matlab/Simulink tool into an ABB IIT product is the Aspect Integrator Platform (AIP). An sample screenshot from the AIP is shown in Fig. 7.

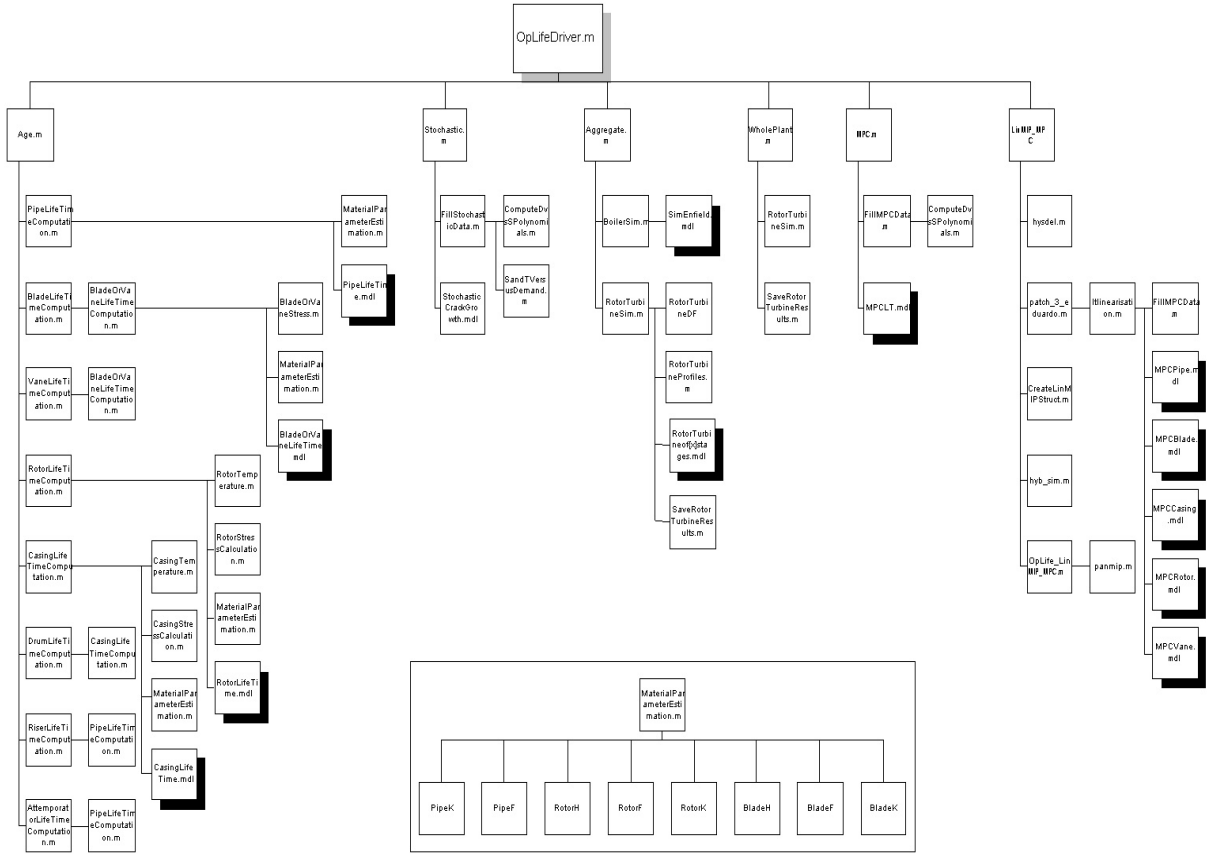

Figure 5: *Overview of the functional structure of the complete tool. The tool consists of a process simulation part* (`aggregate.m` *and* `WholePlant.m`)*, an ageing simulation part* (`Age.m`, `MaterialParameterEstimation.m` *and* `Stochastic.m`) *and an economic optimisation part* (`MPC.m` *and* `LinMP_MPC.m`).

Central to the AIP is the Aspect Object Model which allows a user to present and manipulate information in aconsistent way, features for easily integrating different functions into the AIP are also available. The Aspect Object Model is based on the two ideas of Aspect Objects and Aspects. The objects in the model are the objects the user interacts with. For example, an object can be a turbine, a pipe or a blade. However, the object is simply a container that holds different parts, i.e., aspects of the object. An aspect is a collection of data and operations associated with an object. From a computer science perspective the Aspect Object is a meta-object that describes the aspects that, to a programmer, are more normal OO objects. For more detailed information about the Aspect Object Model and the Integrator Platform, see [6, 7].

In the first phase of the product development, the data structures, the simulation and optimisation routines in Matlab are kept, while the Matlab GUI is replaced by the Industrial^{IT}® platform. Due to the clear interface between the data structures and the GUI, it is relatively straightforward to integrate the Aspect Objects with the Matlab/Simulink code.

The Aspect Integrator Platform provides the possibility of linking to Matlab through an activeX connection and letting Matlab run as a server on the platform. The MATLAB Runtime Server provides one method for pushing variables to the matlab workspace: `PutFullMatrix`. Similarly, the output from `.m` and `.mdl` files should be pushed on the Workspace, from which the aspect system object (`GetFullMatrix`) can read them. The combined Matlab/Simulink and Aspect Integrator Platform approach allows a fast transition from prototype design to a customer product. However, the final product release will port critical Matlab/Simulink models to compiled C++ code for speed improvements and intellectual property protection.

5 Conclusion

Estimating the true costs associated with operating a power plant requires that the cost of damage resulting from operating the plant be monitored and predicted. Once these costs are quantified a tool to support operation is possible. This paper has described how Matlab has been used to develop such a prototype product.

The flexibility and ease of use of the Matlab/ Simulink combination meant that complex models and computation requirements could be developed quickly. The paper concludes by briefly showing how the developed prototype system can be transferred to a commercial product based on

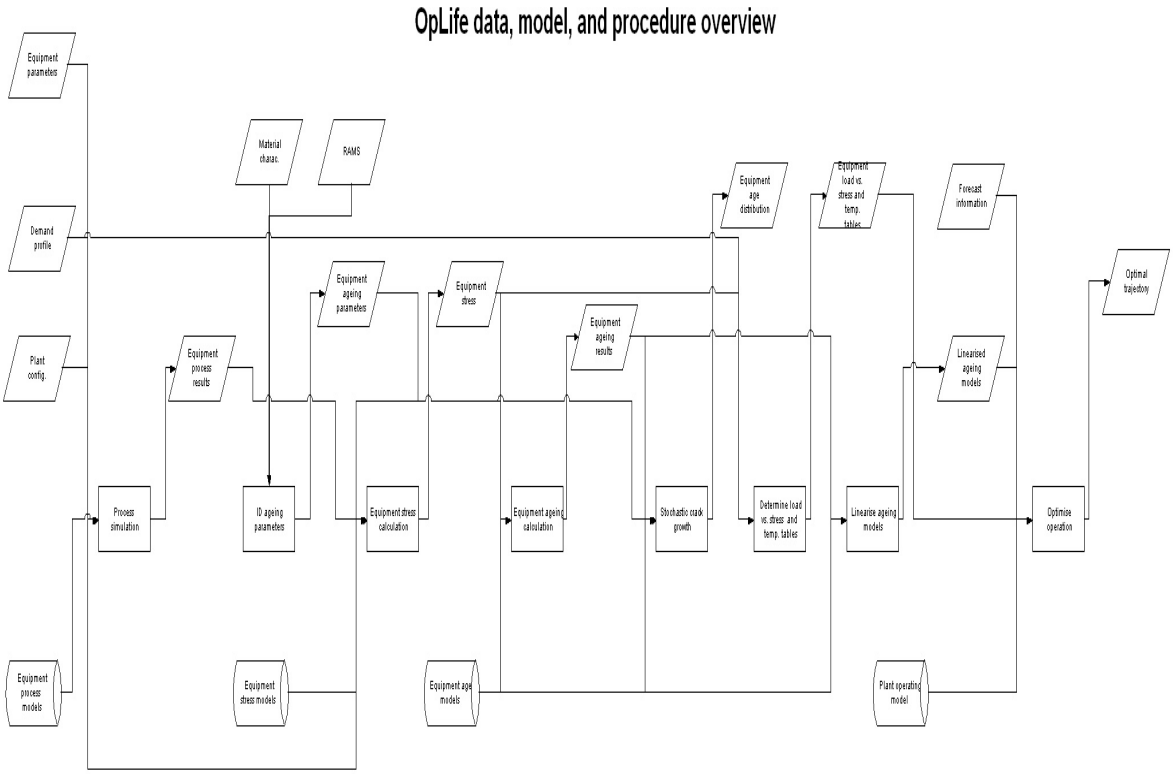

Figure 6: *Flowchart of the complete tool. The diagram shows the execution sequence from input data to the final output data, optimal power output trajectory. The optimal power output is computed for all gas and steam turbines. The cylinders illustrate models, the squares illustrate algorithms and the parallelograms illustrate constant data and lookup tables.*

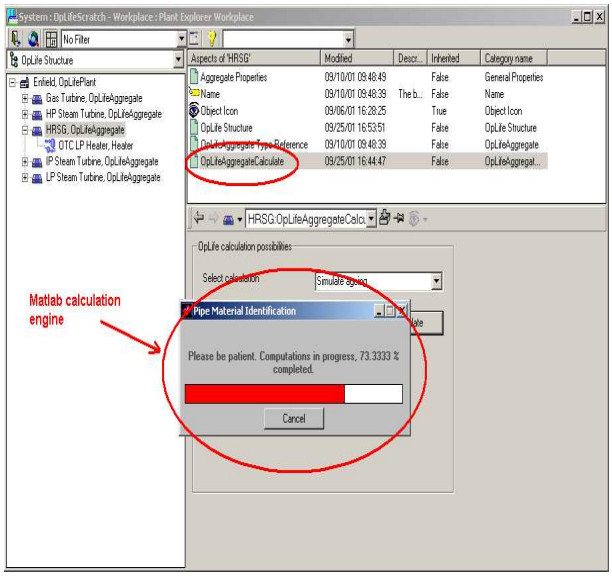

Figure 7: *Migration of Matlab data structure to ABB's Aspect Integrator Platform.*

ABB's new Industrial^{IT}® solution architecture.

References

[1] R. Viswanathan, "Damage Mechanisms and Life Assessment of High-Temperature Components", *ASM International*, 1989.

[2] M.P.Miller, D.C.McDowell, et.al, "A life prediction model for thermomechanical fatigue based on microcrack propagation", *ASTM STP 1186*, 1993, pp 35-49.

[3] C.Cangan, et.al, "Recent developments in the thermomechanical fatigue lifetime prediction of superalloys", *e-JOM*, Vol. 51. No 4. April 1999.

[4] R. Green, "Competition in Generation: The Economic Foundations", *Proceedings of the IEEE*, vol. 88, no. 2, pp 128-139, February 2000.

[5] C. Wang, and S. Shahidehpour, "Optimal Generation Scheduling with Ramping Costs", *IEEE Trans. on Power Systems*, Vol. 10, no. 1, pp 60-67. February 1995.

[6] ABB Automation Products, *Industrial^{IT}®: Aspect Object Architecture Document*, ABB Document Number 3BSE 012 770R101, 2001.

[7] ABB Automation Products, *Industrial^{IT}®: Aspect Integrator Platform, Programmers Guide*, ABB Document Number 3BSE 023 959R101, 2001.

[8] D. Rooke and D. Cartwright, "Compendium of Stress Intensity Factors", *Her Majesty's Stationery Office*, 1976.

[9] E. Camacho and C. Bordons, "Model Predictive Control", Springer 1999.

[10] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints", Automatica, Vol. 35, no. 3, pp 407–427, 1999.

[11] W. Traupel, "Termische Turbomaschinen", Springer Verlag 1988.