

System Level Dynamic Modeling of Fuel Cell Power Plants

Jonas Eborn, Lars Pedersen, Christoph Haugstetter and Shubhro Ghosh

United Technologies Research Center

411 Silver Lane, MS 129-15, East Hartford, CT 06108, USA

Email: {EbornJP, PedersLM, HaugstC, GhoshS}@utrc.utc.com

Abstract

In a joint effort between United Technologies Research Center, UTRC, and the business unit UTC Fuel Cells, UTCFC, system level dynamic models are developed and deployed for the design and analysis of fuel cell based power systems.

The article describes the scope and the challenges within this effort as well as some of the methods and tools used. System level modeling of fuel cell systems is a challenge that few modeling tools can handle due to the heterogeneous nature of the system. The joint work has resulted in a set of proprietary model libraries for unique component designs, building upon publicly and commercially available model libraries and tools.

The model libraries have been used to build system models in both stationary and automotive applications. Examples of models that show the model complexity and some simulation results are given in the paper. Since the system models contain a large number of components, the number of state variables for a full system model usually exceeds 500. Simulation of large, complex systems pose special problems not seen when working with smaller models and really stretch the capabilities of the simulation tools. The paper also discusses some lessons learned in this respect.

1 Introduction

A Proton Exchange Membrane (PEM) fuel cell is an electrochemical device that converts oxygen

and hydrogen to electrical power with water as the only byproduct. Fuel cell based power systems are considered as one of the most promising clean energy sources [4] for both stationary power plants and automotive applications. In the US federal initiative FreedomCar, funding is provided for the development of fuel cell technology with the potential of being considered an alternative to the gasoline car engine.

Fuel cell systems are truly heterogeneous systems involving mechanical, chemical, thermal and electrical systems. This makes system level modeling a significant challenge that very few modeling tools can handle [1]. The challenges consist of both numerical issues and model configuration management issues.

The dynamic modeling team at UTRC and UTC Fuel Cells is using a proprietary model library in Modelica for constructing dynamic system level models for fuel cell systems. The models have been used in the design of the fuel cell control system, including process controls and sequential controls, as well as for systems dynamic analysis and transient performance trade studies.

The primary scope of the fuel cell models is to provide reactant compositions information and information about flows in different components and subsystems. Several different modeling paradigms are available for achieving this goal but in interest of keeping the model order at a minimum a well known approach representing the fuel cell system as a flow network consisting

of pressure drops and lumped parameter volumes has been chosen.

At UTRC and UTCFC system level models have been used in several fuel cell programs in both stationary and automotive applications. Due to the large number of components in the system models, the number of state variables has in general exceeded 500. It is fair to say that this has stretched the capabilities of the simulation tools in several areas.

On-going work on fuel cell component libraries focus on implementing models in a fashion that will minimize model order and maximize model robustness [7]. The lesson learned here is that we cannot expect robust system models unless great care is taken in guaranteeing that all component models are robust.

2 Fuel Cell systems

A fuel cell system consists of a large number of components besides the cell itself. The fuel cell needs streams of oxygen and hydrogen to electrochemically convert into electrical power and water. The oxygen for the fuel cell is normally supplied using air, which is readily available. On the other hand, hydrogen is not available with current energy distribution infrastructures and therefore many fuel cell systems include a Fuel Processing System (FPS). The FPS converts available fuels such as gasoline or natural gas into hydrogen and can easily contain more than half of the components in the full system. Fuel cell power systems usually also include a Thermal Management System (TMS) to manage the heat generated by the PEM Cell Stack Assembly (CSA) and the FPS. To convert the direct current from the CSA into alternating current for the power grid the system also needs to include a Power Conditioning System (PCS).

Figure 1 shows a fuel cell system modeled in Dymola, a dynamic modeling and simulation tool. All lines between subsystems are physical connections, carrying information on pressure, flows and composition. On the outside of the system are triangular input and output

connections, carrying signals to and from the control system.

3 Modeling of heterogeneous systems

Fuel cell systems are truly heterogeneous systems with interactions between mechanical, chemical, thermal and electrical components and subsystems. It also contains a complex flow network near ambient pressure with several recycle flows and strong couplings between subsystems. This makes system level modeling a big challenge that very few modeling tools can handle [1]. The challenges consist of both numerical and model handling issues. Having a model team working concurrently on system models consisting of hundreds of components, several hundred dynamic states and more than 20000 equations is a very complex task. This puts strict requirements on model library structure, version control, etc.

Modelica, an open standard modeling language [3], was conceived from the beginning to be a domain independent language, suitable for modeling of heterogeneous systems. It also has a focus on system dynamics of physical systems. Also, Modelica supports open-code model libraries. The Modelica standard library contains mechanical, electrical and block libraries that may be used for system models. There are also third-party libraries, such as the ThermoFluid library [7] for thermo-hydraulic applications.

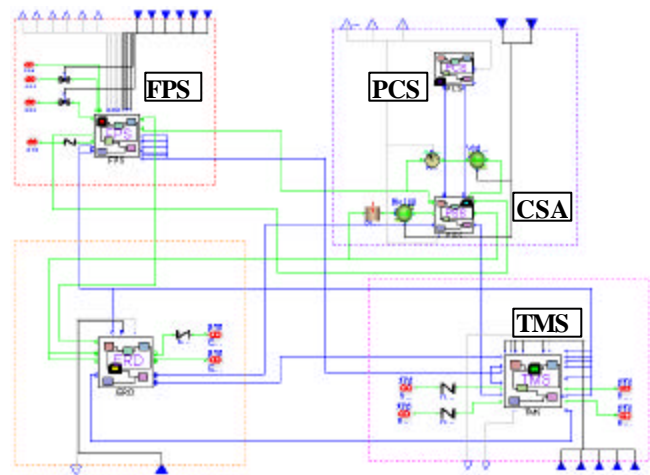


Figure 1: Fuel cell system level model in Dymola, including FPS, CSA, PCS and TMS.

This makes Modelica a very good choice for the heterogeneous system level models necessary to describe fuel cell systems.

4 System level dynamic models

The dynamic modeling team at UTRC and UTCFC has developed a proprietary model library in Modelica for constructing dynamic system level models for fuel cell systems. The model library is based on ThermoFluid, which has an extensible structure suitable for this task. It is also completely based on first-principles models with ideal gas medium properties and lumped volumes or in some cases one-dimensional discretization schemes. The models in the UTCFC library include all major process subsystems: FPS, CSA, TMS, PCS and fuel and air delivery systems. The models have been used in the design of the fuel cell control system, including process controls and sequential controls, as well as for systems dynamic analysis and transient performance trade studies.

System level models are used in several different stages of the development process at UTCFC. Steady-state design models using gPROMS from PSE are used for both conceptual and preliminary design. The steady-state design provides a range of feasible operating points with flow, pressure drop and other design data necessary for the dynamic models.

Models for dynamic analysis and controls structure verification using Dymola are also used in the preliminary design stage, but mainly for evaluating the final design; see the next section for some examples. The dynamic models can then be reused for building models for Hardware-in-the-loop (HIL) testing. The HIL models execute on dSPACE hardware and are used to validate the embedded controls software and hardware before actual hardware tests of the fuel cell system design starts. HIL models are usually based on a simplified version of the dynamic model, obtained through linearization and model order reduction.

In all of these stages physical, first-principles models have several advantages:

- They are reusable and rapidly re-configurable for evaluation of different alternatives in conceptual design.
- They provide physical insight into the component behavior.
- They are capable of giving a realistic dynamic behavior based solely on design data.
- They can be fine-tuned to have better accuracy through validation when experimental component data becomes available.

All of these properties are critical for a rapid development cycle, especially in an emergent technology field such as fuel cell system development.

To facilitate debugging, the system model is constructed in an incremental manner where subsystem models are constructed and tested first, before being added to the system one by one. The system model is then debugged and tuned to match steady state operating conditions. Finally, the model is interfaced to the control system, implemented in Matlab/Simulink. The controlled system model is then used for simulating transient behavior as well as startup and shutdown of the fuel cell power plant.

5 Use of dynamic models at UTCFC

In a collaborative effort between UTRC and UTCFC system level dynamic models (SLDM) have been implemented as a system design tool. They have been used in several fuel cell development programs for both stationary and automotive applications. The use of SLDM for dynamic analysis of systems has many facets in the systems design group at UTCFC.

5.1 Dynamic requirements

The main objective of SLDM is the evaluation of dynamic performance requirements for new fuel cell power plants and its individual components and subsystems. This includes for example load following capability of the power plant. UTCFC's

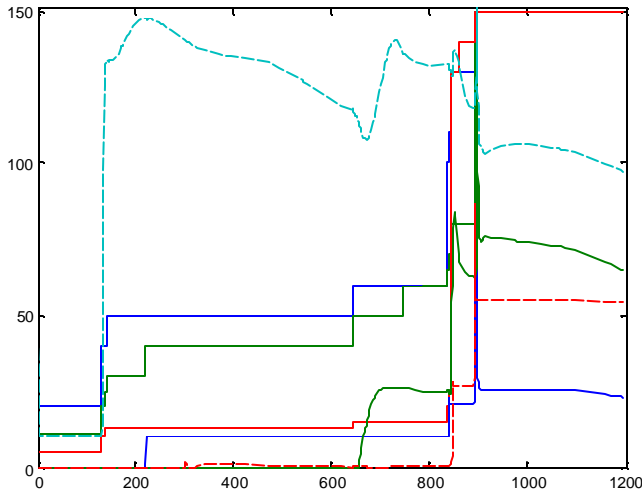


Figure 2: Sample results from a startup simulation for a fuel cell power system. The plot shows the state of three different state machines, fuel, cathode and process air flows and the fuel cell current (dashed). The flows have been rescaled in order to not reveal any proprietary data.

current commercial product, the 200kW PC25 power plant, is mainly used as a source of backup power at critical locations. To meet the demands of this application it is necessary to accommodate instantaneous load changes of up to 80kW.

Emissions requirements is another area where dynamic models are used. Although current standards mainly impose limits on steady-state emissions it is necessary to also evaluate them during load transients and shutdowns. The standards are developing together with the fuel cell technology and dynamic requirements may soon be a part of the standards.

5.2 Sequential controls

Sequential controls verification is another area where dynamic models are necessary. This means stepping through the startup and shutdown sequences as well as any fault conditions to verify the safe operation of the power plant during all stages of the sequence.

The sequential controls for a fuel cell system consist of interacting finite state machines. The state machine for each subsystem consists of 20–50 discrete states. The individual state machines are coded in Matlab/StateFlow and in the full

system model they interact with both the physical plant model and the continuous control loops. A simple example of part of the startup sequence for a stationary fuel cell system is shown in Figure 2. At the end of the simulation, the system has reached state 150 and is on-load, delivering power.

5.3 Continuous controls

Continuous controls analysis includes implementing the complete controls structure of the power plant for verification purposes. Initially, purely physics-based models may only have sufficient accuracy to provide qualitative results, valuable for controls structure decisions and risk mitigation. However, as measurement data becomes available, validated physical models can be used to provide controller parameters for the individual control loops. They can also be used for trouble-shooting as hardware testing begins. Figure 3 shows an example of risk mitigation with dynamic models. Inaccuracy is added to a flow measurement at $t=1400s$ and $1550s$, but the impact is shown to be very small and pose no significant risk.

Simplified versions of the physical system models can be used for rigorous multivariable control design [5] as well as for hardware-in-the-

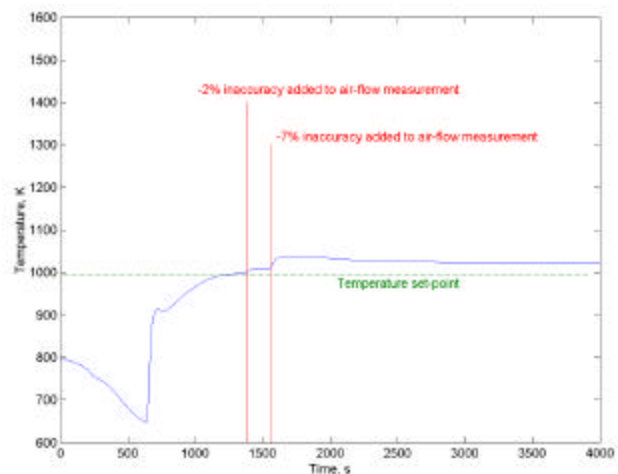


Figure 3: Dynamic models are used e.g. for risk mitigation in sequential and continuous controls. The air blower is turned on at $t=600s$ and then the impact of errors in the flow measurements is investigated.

loop (HIL) verification of the controller hardware and software. HIL testing is currently being implemented as standard work at UTCFC and will be done for every new systems design.

Additional areas of use for dynamic models that are currently investigated in the collaboration between UTCFC and UTRC are fault detection and isolation (FDI) and operations optimization.

6 Model library management

One of the important lessons learned during the course of the SLDM program is the importance of model library management. Model development is a collaborative team effort and requires quality control that has a lot in common with software design and development. Many of the means for obtaining high quality models are already established in the software design industry [2]. Model quality has three equally important aspects:

Robustness, the ability of the models to produce results without numeric problems. This includes initialization of system models with imperfect state knowledge.

Accuracy, the degree of agreement between model results and measurements.

Documentation, brief and accurate descriptions of model assumptions to support use by other team members.

Each of these three aspects can be hard to achieve by themselves and achieving all of them is only possible with a certain degree of discipline within the team. The model development process should support this by entailing:

- Model version control, e.g. with a tool like CVS, concurrent version system.
- Peer code reviews.
- Regression testing, having a set of tests with known results that are run regularly.
- Model qualification testing, standard tests that new models have to pass before being allowed into the official library.

- Parameter management in scripts.
- Naming conventions for components in system models, linked to naming on the process & instrumentation diagrams.

6.1 Model robustness

Initial versions of the UTCFC dynamic model libraries suffered from problems with robustness, slow or non-convergence, solver crashes, etc. This was partly because the development effort used newly developed tools in a new application. But it was also due to the complexity of the system models. The number of state variables in the models quickly grew to exceed 500 dynamic states. It is fair to say that this stretched the capabilities of the simulation tool in a new application area. The issues involved for example creating feasible initial values for a nonlinear flow network with multiple recycle loops. Other issues were major concerns in the simulation of power plant startup with zero- or backflow conditions and operating conditions in temperature and composition very far from the steady-state design point. Issues especially apparent here were eliminating chattering solutions with discrete mode switches in the models, both for flow reversals and for phase changes, and avoiding solver crashes due to singular Jacobians. Problems with singular Jacobians are often caused by semi-empirical expressions for catalytic reaction rates and nonlinear characteristics like the voltage characteristic of a fuel cell. Since these are based on experimental results the expressions are necessarily developed with a limited validity range and often produce undesired results or singularities outside this range.

In subsequent development versions of the model library, these problems have been avoided through several different measures.

- The models are implemented in a code style to avoid singularities, e.g. by guarding division by zero. Even though the validity range for an expression is still limited, the robust operating range can be extended.

- Making use of specific constructs like the Modelica keyword `noEvent`. This is used to avoid unnecessary discrete events for smooth functions containing `if` statements.
- Rigorous Taguchi-style testing of component models [6]. This ensures operability of the models by spanning a wide range of operating conditions in robustness tests. Any potential problems are thus revealed early and do not need to be debugged in large system models.
- Unnecessary complexity should be avoided as far as possible. Almost any model can be used for small examples; robustness problems only become visible in larger, more complex systems. Adhering to Occam's rule and keeping model order at a minimum can avoid problems. The structure of the resulting equations should also be checked for non-linear algebraic equation systems that sometimes arise from simplifying assumptions.

7 Conclusions

Going forward, the primary focus will be on further developing the component models.

includIn systemsgovheneed byparntialdiffertential equations.

nvironemens thas eaobl2s apoid construction and of systemldevet models. Thg

robustllyovhe thefuill operating(nvvelore of the) Tj 0 -13.5 TD 0.0657 Tc 29938

so, Chari McCready,