VIRTUAL ADAPT - A Simulator for ADAPT

1. Introduction

Motivation:

The Advanced Diagnostic and Prognostic Testbed (ADAPT), developed at NASA Ames Research Center, is functionally representative of an electrical power system (EPS) on an exploration vehicle, and has been developed to:

(i) Serve as a technology-neutral basis for testing and evaluating software and hardware diagnostic systems,

(ii) Allow accelerated testing of diagnostic algorithms by manually or algorithmically inserting faults,

(iii) Provide a real-world physical system such that issues that might be disregarded in smaller-scale experiments and simulations are exposed,

(iv) Act as a stepping stone between pure research and deployment in aerospace systems, thus creating a concrete path to maturing diagnostic technologies, and

(v) Develop analytical methods and software architectures in support of the above goals.

The ADAPT hardware includes components that can generate, store, distribute, and monitor electrical power. The EPS can deliver AC (Alternating Current) and DC (Direct Current) power to loads. A data acquisition and control system sends commands to and receives data from the EPS. The testbed operator stations are integrated into a software architecture that allows for nominal and faulty operations of the EPS, and includes a system for logging all relevant data to assess the performance of the health management applications.

The ADAPT Hardware:

Figure 1 depicts ADAPT's major system components and their interconnections. Two power generation sources are connected to three sets of batteries, which in turn supply two load banks. Each load bank has provision for 6 AC loads and 2 DC loads. To be more specific, ADAPT consists of the following three classes of components – power generation, power storage, and power distribution.

(i) *Power Generation:* The two sources of power generation include two battery chargers. The battery chargers are connected to appropriate wall outlets through relays. The two power generation sources can be interchangeably connected to the three batteries. Hardware relay logic prevents connecting one charge source to more than one battery at the same time, and from connecting one charging circuit to another charging circuit.

(ii) *Power Storage:* Three sets of batteries are used to store energy for operation of the loads. Each "battery" consists of two 12-volt sealed lead acid batteries connected in series to produce a 24-volt output. Two battery sets are rated at 100 Amp-hrs and the third set is rated at 50 Amp-hrs. The batteries and the main circuit breakers are placed in a ventilated cabinet that is physically separated from the equipment racks; however, the switches for connecting the batteries to the upstream chargers or downstream loads are located in the equipment racks.

(iii) *Power Distribution:* Electromechanical relays are used to route the power from the sources to the batteries; and from the batteries to the AC and DC loads. All relays are of the normally-open type. An inverter converts the 24-volt DC battery input to a 120-volt r.m.s. AC output. Circuit breakers are located at various points in the distribution network to prevent overcurrents from causing unintended damage to the system components.

ADAPT Software:

Unlike many other testbeds, the primary articles under test in ADAPT are the health management systems, and not the physical devices of the testbed. To operate the testbed in a way that facilitates studying the performance of the health management technologies, the following operator roles are defined:

(i) *User:* The user simulates the role of a crew member or pilot operating and maintaining the testbed subsystems

(ii) *Antagonist:* The antagonist injects faults into the subsystem either manually, remotely through the Antagonist console, or automatically through software scripts.

(iii) *Observer:* The observer records the experiment data and notes how the User responds to the faults injected by the Antagonist. The Observer also serves as the safety officer during all tests and can initiate an emergency stop.

During an experiment the User is responsible for controlling and monitoring the EPS and any attached loads that are required to accomplish a mission. The Antagonist disrupts system operations by injecting one or more faults unbeknownst to the User. The User may use output from a health management application (test article) to determine the state of the system and choose an appropriate recovery action. The Observer records the interactions and measures the effectiveness of the test article.

VIRTUAL ADAPT:

VIRTUAL ADAPT is a high-fidelity, Matlab® Simulink®-based simulation testbed that emulates the ADAPT hardware for running offline health management experiments. This simulation testbed models all components of the ADAPT hardware within the power storage and power distribution subsystems. The physical components of the testbed, i.e., the batteries, relays, and the loads, are replaced by simulation modules that generate the same dynamic behaviors as the hardware test bed. Figure 2 shows the parts of the ADAPT hardware that are modeled in VIRTUAL ADAPT.

This distribution contains *VirtualADAPT.mdl*, a standalone Simulink model that does not interface with the messaging server.

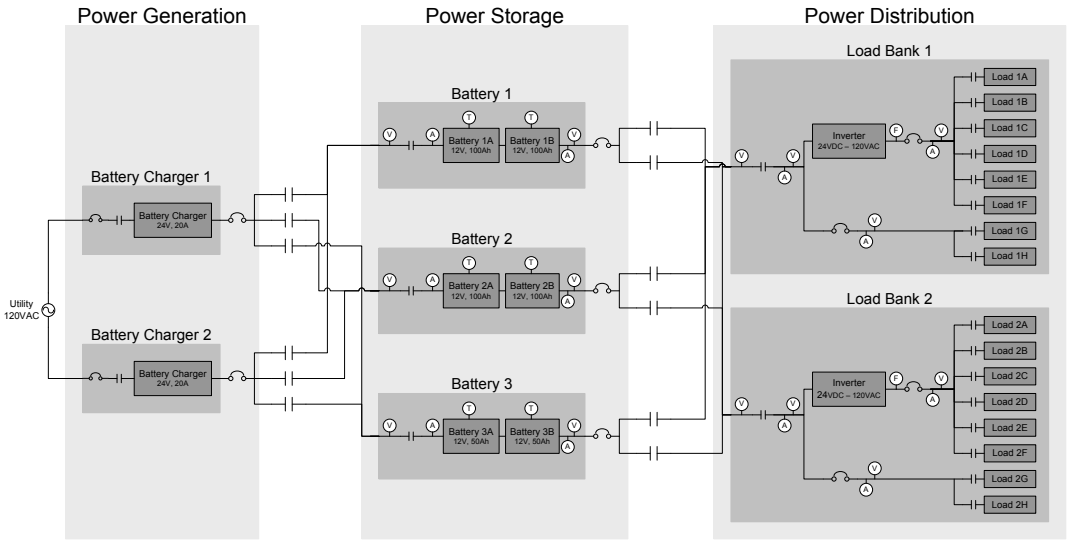


Figure1: ADAPT System Schematic

2. Modeling

In order to mirror the real testbed, we have addressed several issues that include (i) one-to-one component modeling, (ii) replicating the dynamic behavior of the hardware components, (iii) matching the possible configurations, and, (iv) facilitating the running of diagnosis and prognosis experiments.

The approach to component-oriented compositional modeling is based on a top-down process, where we first capture the structural description of the system in terms of the components and their connectivity. Component models replicate the component's dynamic behaviors for different configurations. The connectivity relations, represented by energy and signal links, capture the interaction pathways between the components. Complex systems, such as the power distribution system, may operate in multiple configurations. The ability to change from one configuration to another is modeled by switching elements. Sets of components can also be grouped together to define subsystems. The VIRTUAL ADAPT testbed component library includes models for the batteries, inverters, loads, relays, circuit breakers, and sensors that exist in the current ADAPT hardware testbed. For experimental purposes and "what if" analyses, new components can be added to the library by modifying existing component models or by creating new ones. System models are built by creating configurations of component models.

Many ADAPT testbed components are physical processes that exhibit hybrid behaviors, i.e., mixed continuous and discrete behaviors. These components are modeled as Hybrid Bond Graph (HBG) fragments. HBGs extend the bond graph modeling language by introducing junctions that can switch on and off. Bond graphs are a domain independent, topological modeling language based on the conservation of energy and continuity of power. Nodes in a bond graph model include energy storage elements, capacitors (C) and inertias (I), energy dissipation elements, resistors (R), energy transformation elements, gyrators (GY) and transformers (TF), and input-output elements, sources of effort (Se) and sources of flow (Sf). The connecting edges, called bonds, represent the energy exchange pathways between these elements. Each bond is associated with two generic variables: effort and flow. The edges of the bond graph, called bonds, represent the energy exchange pathways between the connected nodes. Each bond has two associated variables: effort and flow, and the product of effort and flow is power, i.e., the rate of energy transfer between the connected elements. The effort and flow variables take on particular values in specified domains, e.g., voltage and current in the electrical domain, and force and velocity in the mechanical domain. Components and subsystems in bond graph models are interconnected through idealized lossless 0– and 1–junctions. For a 0–junction, which represents a parallel connection, the effort values of all incident bonds are equal, and the sum of the flow values is zero. For a 1–junction, which represents a series connection, the flow values are equal and the sum of the effort values is 0. Non-linear system behaviors are modeled by making the bond graph element parameters algebraic functions of other system variables.

HBGs extend bond graphs using a two state automata model (the two states are ON and OFF) that controls the switching of junctions between on and off. Using switching functions, the modeler can capture discrete changes in system configuration, such as the turning on and off of a relay. The switching of junctions can be controlled or autonomous. The on-off transitions for controlled junctions are determined by external signals, e.g., a controller input, whereas the on-off transitions for autonomous junctions depend on system variables.

Building complete models for a system requires detailed knowledge of the system configuration and component behaviors, as well as component parameters. This knowledge has been obtained using information from device manuals, research papers, and experimental data collected during system operations. Unknown system parameters and functional relations associated with the models have been estimated using system identification techniques. Model validation has been performed by comparing simulated behaviors with data collected from experimental runs on ADAPT.

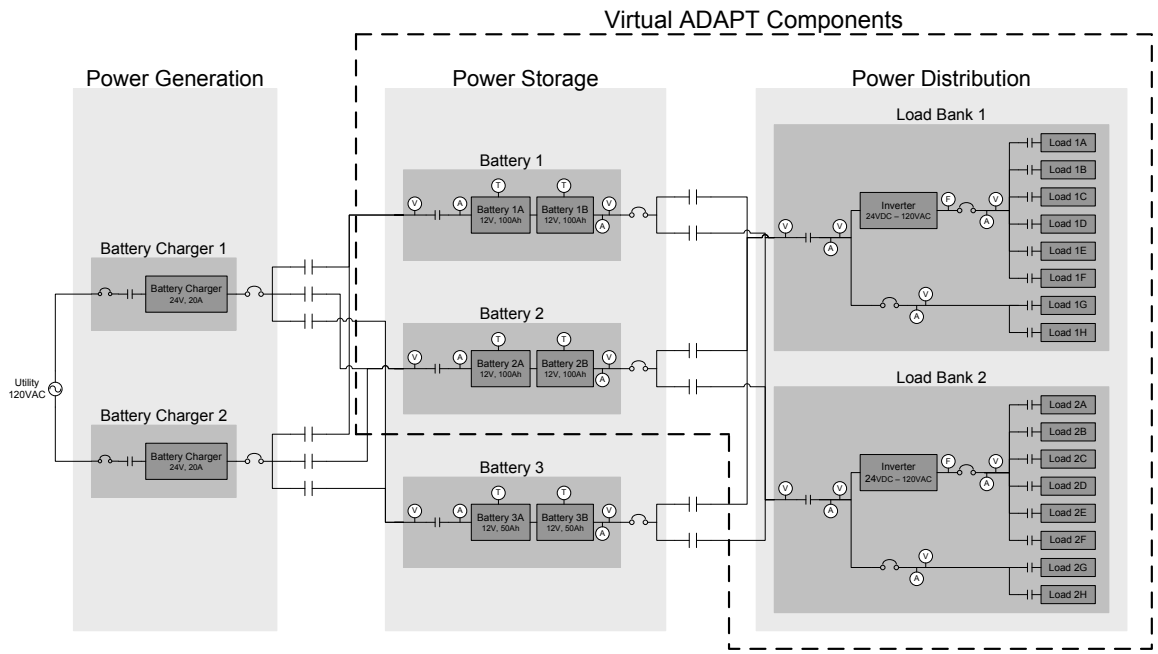


Figure 2: Parts modeled in VIRTUAL ADAPT

3. VirtualADAPT: User Guide

Initializing the model:

Before the model is run, one needs to initialize some component properties, such as battery voltage, relay positions, and so on. The settings of the initial configuration and nominal load settings are given below.

Configuration Settings:

- All circuit breakers closed.
- All relays open.
- Batteries 1 and 2 set to 25 Volts..

Load Settings:

- LGT6: All 3 light bulbs are turned on
- FAN1: Speed setting 2
- FAN3: Speed setting high
- LGT8: Light bulb on
- PMP2: The valve turned to the marked nominal setting. The flow sensor reads approximately 186 units at the nominal setting.
- LGT4: Light bulb on
- NEMO: All DC loads on except the buzzer
- PMP1: The valve turned to the marked nominal setting. The flow sensor reads approximately 192 units at the nominal setting.
- LGT2: Light bulb on
- FAN4: Speed setting high
- LGT9: Light bulb on
- LGT7: All 3 light bulbs are turned on
- FAN2: Speed setting 2
- DC2: Set to 2 Amps

Note that the model parameters have been identified for the above load settings.

Running the simulation:

To run the Simulink model, simply open the created model and press the "run" button. The data structure script, "VirtualADAPTDataStructure.m", must be in the same directory as the model for it to execute successfully. Once the simulation starts, the user may change the switch inputs to close or open relays and controlled circuit breakers to bring the simulation to the desired mode of operation.

Initializing the Model and Injecting Faults:

Any parameter values can be set by opening that part of the model, double-clicking on the component, and setting the parameters in the component's mask, as shown in Figure 3. For example, to initialize the battery voltage, look for 'VirtualADAPT/VirtualADAPTv1/PowerStorage/Battery1Subsystem/Battery1/CircuitEquivalent/C0' in the GUI for Battery 1. Alternatively, the function "initBatteries.m" is provided, so that, calling "initBatteries(25,26);" sets the initial voltage of Battery 1 to 25 Volts and of Battery 2 to 26 Volts.

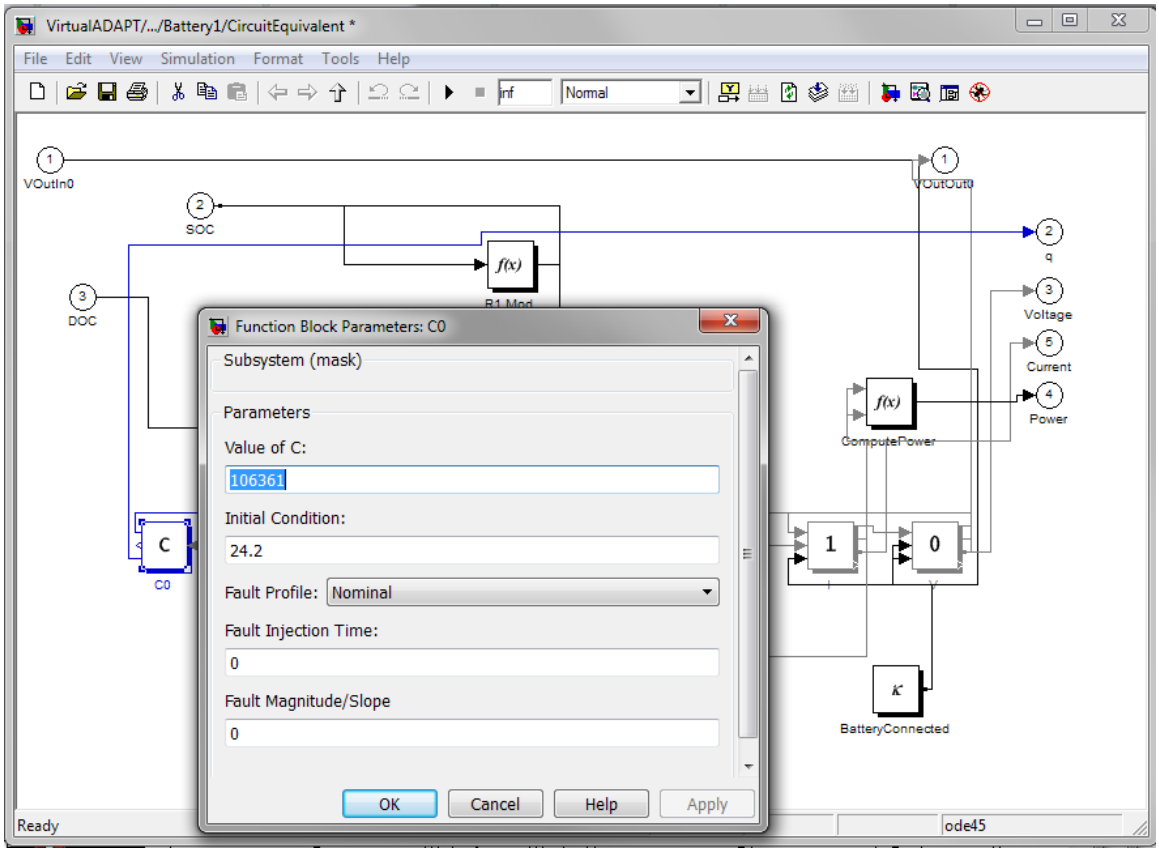


Figure 3: Setting parameter values using component masks.

Faults can be injected and parameter values set by using the fault injection interface shown in Figure 4. To inject a fault, first select the component where the fault is injected from the "Component" drop down menu. Based on which component is selected, applicable fault profiles are presented using the "Fault" drop-down menu. Select the desired fault profile, as well as the fault injection time and the fault magnitude in the "Injection Time" and "Magnitude" fields, and hit the "Add Fault" button. To remove a fault, select the appropriate row corresponding to the fault that is to be removed, and hit the "Remove Fault" button. The "Remove All Faults" button removes all faults, thereby resetting the system to nominal. Hit the "Save Configuration" button to save the current settings of the interface to a .mat file. Hit the "Load Configuration" button to load previously saved settings.

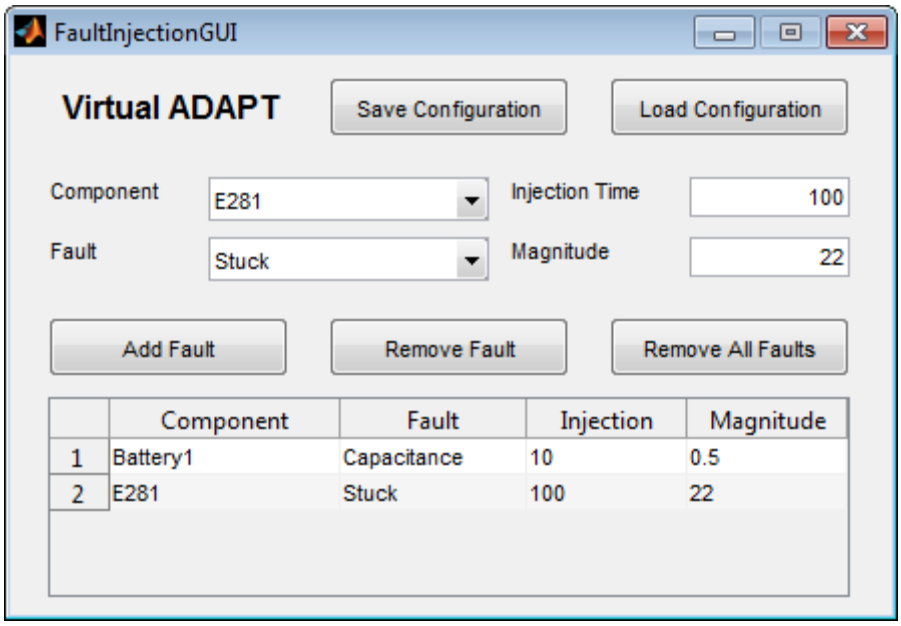


Figure 4: Fault Injection Interface

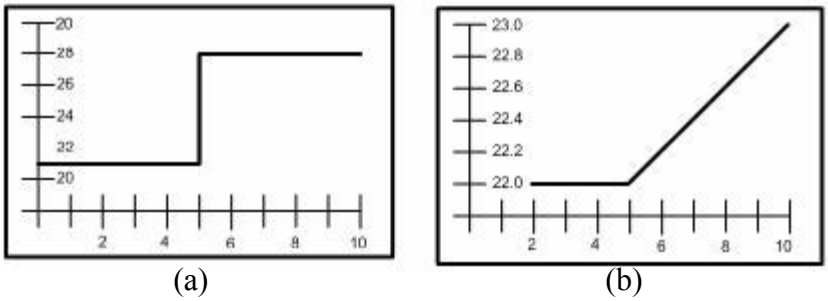
The tables below describe the generic fault modes that can be set, the meaning of the magnitude value for these modes, and the meaning of the fault injection time.

Fault Mode	
Nominal	No fault.
Abrupt	Instantaneous (step) change in the system parameter value.
Incipient	Gradually changing parameter value over time. Approximated as a linearly changing system parameter over time with a constant slope.
Bias	Addition of some value to the system parameter causing an instantaneous (step) change in the system parameter value.
StuckAt	Instantaneous (step) jump of system parameter to a fixed value.

Fault Magnitude	
For Abrupt Faults	The multiplicative change in the magnitude of the nominal parameter value (can be positive or negative). The faulty value is the nominal parameter value plus one, multiplied by the entered value.
For Incipient Faults	The rate of change in the magnitude of the nominal parameter value (can be positive or negative) over time. The slope is the entered value.
For Bias faults	The additive change in the magnitude of the nominal parameter value (can be positive or negative). The entered value is added to the nominal parameter value.
For Stuck faults	The value the parameter gets stuck at. The nominal parameter value becomes this value and stays at this value.

Fault Injection Time
The time instance when the fault is injected in the system

Figure 3 below illustrates the abrupt and incipient fault profiles.



(a) (b)

Figure 3: (a) Abrupt fault profile (b) Incipient fault profile

1. References:

- I. Roychoudhury, M. Daigle, G. Biswas, and X. Koutsoukos, "Efficient Simulation of Hybrid Systems: A Hybrid Bond Graph Approach," SIMULATION: Transactions of the Society for Modeling and Simulation International, vol. 87, no. 6, pp. 467-498, June 2011.

- S. Poll, A. Patterson-Hine, J. Camisa, D. Garcia, D. Hall, C. Lee, O. Mengshoel, C. Neukom, D. Nishikawa, J. Ossenfort, A. Sweet, S. Yentus, I. Roychoudhury, M. Daigle, G. Biswas, and X. Koutsoukos, "Advanced Diagnostics and Prognostics Testbed," *18th International Workshop on Principles of Diagnosis*, pp. 178-185, May 2007.

- S. Poll, A. Patterson-Hine, J. Camisa, D. Nishikawa, L. Spirkovska, D. Garcia, D. Hall, C. Neukom, A. Sweet, S. Yentus, C. Lee, J. Ossenfort, I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, and R. Lutz, "Evaluation, Selection, and Application of Model-Based Diagnosis Tools and Approaches," *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, May 2007.

- M. Daigle, I. Roychoudhury, G. Biswas, and X. Koutsoukos, "Efficient Simulation of Hybrid Models Represented as Hybrid Bond Graphs," *Hybrid Systems: Computation and Control (HSCC 2007),* vol. 4416, pp. 680-683, Apr 2007. (short paper)

- I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, and Pieter J. Mosterman, "A Method for Efficient Simulation of Hybrid Bond Graphs," *International Conference on Bond Graph Modeling (ICBGM 2007),* pp. 177-184, Jan 2007.

- D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, Systems Dynamics: Modeling and Simulation of Mechatronic Systems, 3rd ed. New York, NY: John Wiley & Sons, Inc., 2000.