A Model-Based Approach for Safety Assessment Processes

Peter Bunus[(1)(2)]

[(1)]*Department of Computer and Information Science*
Linköping University
SE 581-32, Linköping, SWEDEN
Email: peter.bunus@liu.se

[(2)]*SystemicsCode Nordic AB*
Allmogegatan 104, SE 583-33, Linköping, SWEDEN
Email: peter.bunus@systemicscode.com

ABSTRACT

In recent years, model-based diagnostics reasoning systems have provided a major advance in fault isolation and reduction of repair time contributing to the reduction of maintenance cost of automotive and avionics systems. In this paper we show how model-based diagnosis can be employed for system validation and for design of avionics/aerospace systems with improved readiness, maintainability and availability. We illustrate the application of our approach on a satellite power generation, storage and distribution system test bed (EPS) with real hardware from NASA Ames called ADAPT (Advanced Diagnostics and Prognostics Testbed).

1. INTRODUCTION

With increased system complexity in recent years, almost every system under deployment is exposed to component failures or is under the risk of suffering a major breakdown under its lifetime. The necessity of guarantee the safety of engineering system and the growing importance of lowering the maintenance and repair cost demands for a closer integration of safety assessment tasks in the entire design process and reuse of product related information through all the product development cycle. Existing safety assessment standards offer general guidelines for integration of the safety life cycle into the product life cycle. We name a few of most widely used safety assessment procedures and reliability prediction standards used in practice:

- MIL-STD-882D "System Safety Program Requirements"/"Standard Practice for System Safety" (US Department of Defense 2000 [25]).
- MIL-HDBK-217F “Military Handbook - Reliability Predication of Electronic Equipment” US Department of Defense 1995 [24]
- FIDES Guide 2009. FIDES is a global methodology for reliability engineering in electronics and contains a reliability guide and a reliability process control and audit guide (Airbus et al. 2009 [1]).
- AS 61508 "Functional safety of electrical/electronic/programmable electronic safety-related systems"
- RTCA/DO-178B "Software Considerations in Airborne Systems and Equipment Certification".(RTCA [20])
- Def Stan 00-56 Safety Management Requirements for Defense Systems
- +SAFE - A Safety Extension to CMMI
- SAE ARP 4754 "Certification Considerations for Highly-Integrated or Complex Aircraft Systems" (SAE 1996 [22]).
- SAE ARP 4761 "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment" (SAE 1996 [23])
- Def(Aust) 5679 "The Procurement Of Computer-based Safety Critical Systems"

Most of the standards offer general guidelines for integration of the safety life cycle into the product life cycle. However all the standards and guidelines will require a customization in order to adapt it to the current practices and workflow of an organization, with the proviso that the prescribed safety requirements will be met.

The safety assessment will generally go through several main stages as functional hazard analyses (FHA), preliminary fault tree analysis (Prelim FTA), common cause analysis (CCA) or failure mode and effect analysis (FMEA) to derive safety requirements before the engineering process takes over to realize the system. Fig. 1 depicts the SAE ARP 4754 safety assessment diagram that illustrates some of these safety assessment analysis and the interrelationships between them in the in the framework of the certification process of for highly integrated and complex aircraft systems. It should be noticed the complex interrelationship between these types of safety analysis performed at different stages of the design process of an aircraft. While different software tools are used to address these aspects, none of the tools available on the market addresses the whole safety assessment cycle nor can provide a mode-based technology for supporting the assessment process in the avionics and aerospace industry. Moreover there is a lack of integration of software testing

and verification techniques into the safety assessment process. Software modules are often tested separately from the hardware modules or at the end of the development cycle when it is extremely hard to change the design of the system.

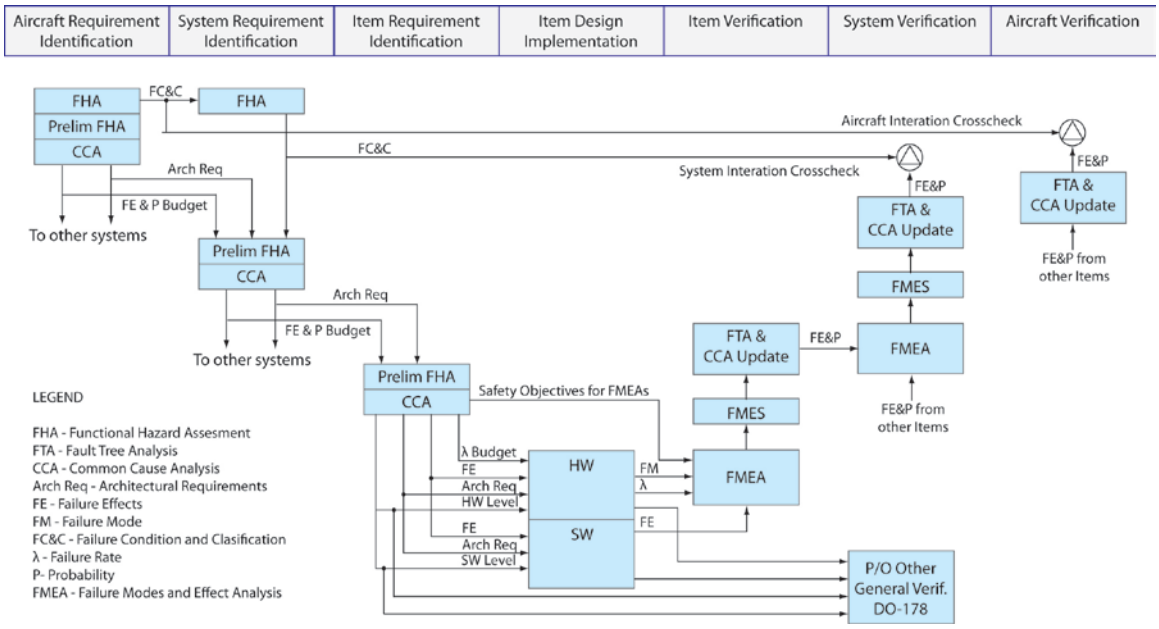

Fig. 1. The ARP 4754 Safety Assessment Diagram.

The multitude of tasks that need to be performed in the avionics safety assessment process would require the usage of various tools and methodologies in different stages of this process. One of the main goals of the model-based safety assessment process proposed in this paper is to provide a workflow between these tools in such a way that the safety assessment process will be supported continuously and information should be passed from one stage to another without manual transitions. Model-based engineering and validation approaches must be exploited to efficiently support the design for safety and diagnosability approach and the rigorous assessment of these properties as well as supporting the traceability during the safety assessment process.

The rest of the paper is organized as follows: In Section 2, we give a brief description of the principles of model-based diagnosis. Section 3 provides a brief description of our prototype tool called Systemics Analyst that supports model-based safety assessment processe. In Section 4, we describe the satellite power system from NASA Ames together with the corresponding model used for diagnosis purposes.. In Section 5 we illustrate how diagnostics can be performed together with some failure scenarios. Finally, Section 6 presents a summary of the paper, the future work and our conclusions.

2. PRINCIPLES OF MODEL-BASED DIAGNOSIS

In the last decade, model-based technology in diagnosis matured so far that it was transferred from academic research into real applications. It provides an alternative to more traditional techniques based on experience, such as rule-based reasoning systems or case-based reasoning (CBR) systems. Early model-based diagnosis tools include MDS (Mauss et al. 2000 [16]), RAZ'R (Sachenbacher et al. 2000 [21]), Livingstone 2 (Hayden et al. 2004 [8]), LYDIA (Feldman et al. 2006 [5]), DSI Express (Gould 2004 [6]), TEAMS-QSI (Deb et al. 1998 [3]) or RODON (Lunde et al. 2006 [13]).

The basic principle of model-based diagnosis consists in comparing the actual behavior of a system, as it is observed, with the predicted behavior of the system given by a corresponding model. A discrepancy between the observed behavior of the real system and the behavior predicted by the model is a clear indication that a failure is present in the system. Diagnosis is a two-stage process: in the first stage, the error should be detected and located in the model, and in the second stage, an explanation for that error needs to be provided. Diagnoses are usually performed by analyzing the deviations between the nominal (fault free) behavior of the system and the measured or observed behavior of the malfunctioning system. As a general rule, the models are built to enable the identification of failed Line Replaceable Units (LRUs). Once a model of the real system is built, simulation or prediction can be performed on the model. The predicted behavior, which is the result of the simulation, can then be compared with the observed behavior of the real system. This comparison is usually done by a reasoning that is able to detect discrepancies and also to generate and propose corrective actions that need to be performed on the real system to repair the identified fault.

In Fig. 2, a model of a real system (a satellite system) is depicted at the lower left corner. It might contain, for example, the behavior of the mechanical components incorporated in the orientation system of the solar panel, the behavior of the electronic telemetry system, the electric power system, or all of them. Note that, like all models, the

model is only an abstraction of the real system (depicted at the upper left corner) and can be incomplete. The granularity of the model and the amount of information and behavior captured into it will directly influence the method employed by the reasoning engine as well as the precision of the diagnostic process.

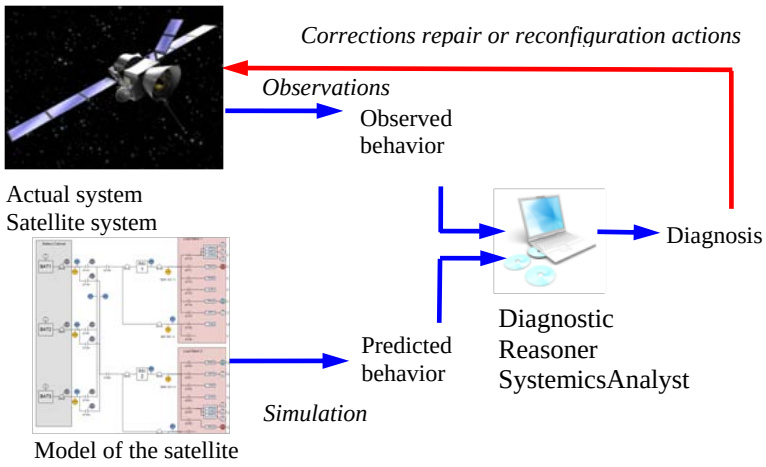

Fig. 2. ***Basic Principles of Model-Based Diagnostic System***

As a general rule, the models are built to enable the identification of failed Line Replaceable Units (LRUs). Once a model of the real system is built, simulation or prediction can be performed on the model. The predicted behavior, which is the result of the simulation, can then be compared with the observed behavior of the real system. This comparison is usually done by a reasoning engine that is able to detect discrepancies and also to generate and propose corrective actions that need to be performed on the real system to repair the identified fault. We should note that the process of diagnosis (incorporated in the diagnostic reasoner) is separated from the knowledge about the system under diagnosis (the model). This ensures that the model can be reused for other purposes as well, such as optimization, reliability analysis, automatic generation of failure mode and effect analysis FMEA, etc.

The ideas of model-based reasoning can be illustrated by the following simple multiplier-adder circuit taken from Kleer and Kurien 2003 [9].

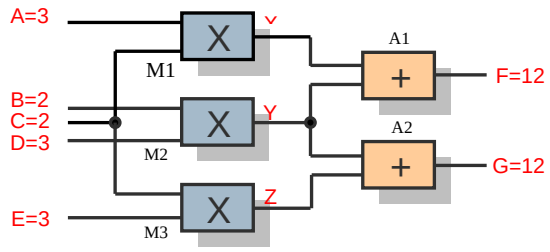

Fig. 3. Multiplier-Adder Circuit

The inputs to the system are `A`, `B`, `C`, `D` and `E`, while the outputs of the system are `F` and `G`. `X`, `Y` and `Z` are intermediary probing points in the system. If requested, the user can perform an observation and measure the value in these points. If the system is supplied with the following input set: `A=3`, `B=2`, `C=2`, `D=3`, `E=3`, then the expected output should be `F=12` and `G=12`. The output is calculated by using a deduction-based reasoning, first computing the intermediate results `X=6`, `Y=6`, and `Z=6` at the output of the multiplier gates, which then are becoming inputs to the adder gates.

For computing the results, a simple inference engine like the ones used by most simulation environments is enough to perform the calculations. Note that the behavior of the components is formulated in terms of relations between model variables which are called constraints. In general, constraints are not directed (unlike assignments), and may be of various nature, e.g. equations, inequalities, or logical relations. A *conflict* is any discrepancy or difference between the prediction made by the inference engine and the observed behavior. Now let us suppose that by observing the real system one notices that the output value of `F` is `10`, which is different from the expected calculated value of `12`. In our case, the observation `F=10` leads to a conflict that will be the triggering point for the diagnostic engine to indicate that something is wrong. It means that one of the components in the system does not work correctly, in other words: there is a contradiction between the assumptions and the observed behavior. In the next step, the diagnostic reasoner should find an explanation for the contradictory observation.

A possible explanation of this behavior might be that adder `A1` or the multiplier `M1` is defective which might cause the defective output `F=10` (as depicted in Fig. 4).

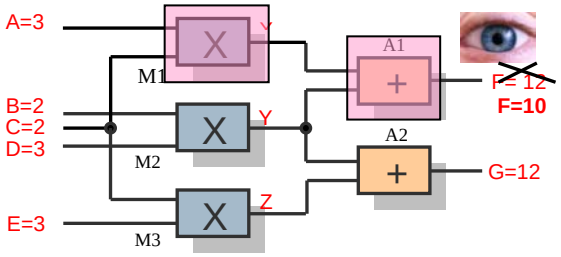

Fig. 4. A defective adder `A1` or multiplier `M1` might explain the wrongly computed value `F=10`

A wrongly computed output by the adder `A1` might be due to the fact that the adder itself is defective, in which case we can add the adder to the possible list of candidates, or due to the wrong inputs received by the adder. It should be noted that the inputs of adder `A1` are the outputs of multiplier `M1` and `M2`. The output of `M2` is also input to the second adder `A2` that computes the value of `G`. Since the value of `G` was correctly computed, we can exclude `M2` from the suspected candidates. A defective `M2` would certainly influence the computed value of `G`. Therefore, with the current information about the system we can conclude that either `M1` or `A1` is defective.

So far, we have considered only single faults in our multiplier-adder circuit. Considering multiple simultaneous faults will give us a new set of candidates, as depicted in Fig. 5.

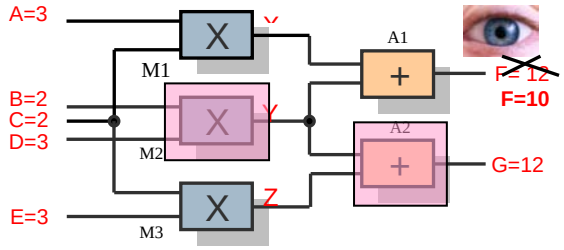

Fig. 5. Multiple fault: Both `M2` and `A2` are defective

Let us analyze the set of multiple fault candidates shown in Fig. 5 and see if it is consistent with the observation. The multiplier `M2` is defective, which means that it will produce a wrong output, which will be input to the adder `A1`. With a wrong input, the adder `A1` will produce a wrong output, which might explain that we observe `F = 10`. Moreover, the output of `M2` is also input to the adder `A2`, which is also defective. It might happen that the defect inside `A2` compensates the defect of the multiplier `M2`, and by chance, it produces a good value for the output `G`. With the given information, this is what we can conclude about the system.

This reasoning method is called abductive reasoning. It starts from a set of accepted facts and infers their most likely explanations in the form of hypotheses. If at a later time, new evidence emerges which disagrees with a hypothesis, this hypothesis will be proven false and must be discarded. In our example, the hypothesis "M2 and A2 are faulty simultaneously" might be disproved by an additional measurement of the value Y = 6. But without additional information about the system, it is a valid hypothesis that explains the abnormal observation. By using the same type of reasoning, more hypotheses can be advanced, e.g. “`M2` and `M3` are simultaneously faulty”.

3. THE SYSTEMICS ANALYST MODEL-BASED SAFETY ASSESMENT SYSTEM

Systemics Analyst is a Model Based Diagnostic (MBD) engine based on the principles of the General Diagnostic Engine (GDE) as described by de Kleer and Williams 1987 [10]. These principles have been further advanced and tailored to the Systemics Analyst applications to increase efficiency and diagnostic power. Systemics Analyst uses contradictions (conflicts) between the simulated and the observed behavior to generate hypotheses about possible causes for the observed behavior. If the model contains failure modes besides the nominal behavior, these can be used to verify the hypotheses, which speed up the diagnostic process and improve the results. But even if not all imaginable fault modes of a component can be included in the model, it is possible to define an unknown fault mode which summarizes all other possibilities of the component to fail. Thus Systemics Analyst can find unexpected or unknown faults as well.

The basic fault detection and isolation method is conflict-driven. All the available signals are fed into the model. They are then propagated across the model from the points of “value injection” or stimulation to simulate or predict the nominal behavior of the system. If during this simulation contradictions between the observed and the simulated behavior are detected, then the system is no longer consistent with the model and is assumed to be defective. The contradictions between calculated or observed values are called conflicts. They are the starting point for the diagnostic process. This process first generates candidates which may explain the deviation between the observed symptom vector and the behavior predicted by the model. If the simulation result with such a candidate is consistent with the observed values we have a possible explanation for the malfunction or a diagnosis. The results of this diagnostic process are printed to a console and in addition the suspect components are highlighted in the model visualizer if the graphical user interface is activated.

The existence of a modeling language that can be used for capturing model knowledge for diagnostics purposes is central for model-based-diagnosis and for supporting an unified safety assessment process. Early model-based diagnosis

systems used traditional general purpose programming languages for specifying models. However, in some aspects, general purpose programming languages are inadequate to the task of formalizing significant domains of engineering practice. They lack the expressiveness and power of abstraction required by engineers. For specifying the diagnostics models Systemics Analyst uses a declarative equation-based language called Modelica (Modelica Association 2007 [2]). Modelica has the following main characteristics:

- Acausal modeling based on mathematical equations to describe the behavior of the system.
- Multi-domain modeling capability, which gives the user the possibility to combine electrical, mechanical, thermodynamic, hydraulic, etc., model components within the same application model.
- A general type system that unifies object-orientation, multiple inheritance and generic templates within a single class construct. This facilitates the reuse of components and evolution of models.
- A strong software component model, with constructs for creating and connecting components.

At the lowest level of the Modelica language, equations are used to describe the relations between the quantities of a model and to define the behavior of the class.

4. THE NASA ADAPT SYSTEM

Assessment and comparison of different diagnostics technologies can be difficult. To facilitate this task the researchers at NASA Ames Research Center have developed the Advanced Diagnostics and Prognostics testbed called ADAPT (Poll et al. 2007 [19]). The testbed acts as a common platform where different diagnostics tools and technologies, so called test articles, can compete against each other on equal conditions. To achieve this, ADAPT consists of a controlled and monitored environment where faults are injected into the system in a controlled manner and the performance of the test article is carefully monitored. The hardware of the testbed is an electrical power system (EPS) of a space exploration vehicle. The testbed is located in a laboratory at the NASA Ames Research Center.

The ADAPT system consists of three major modules: a power generation unit, a power storage unit and a power distribution unit. The interconnections among the different units is depicted in Fig. 6 (picture reproduced from Poll et al. 2007 [19]).

The *Power Generation* unit can charge the batteries located in the *Power Storage Unit* with the help of two battery chargers and a photovoltaic unit (solar panel). The power generation unit is divided into six subsystems: the solar panel unit, the battery charger panel, the protection and enable panel and three battery-charge selection panels. The power storage unit contains three battery packs and several relays that control the connections between the load bank and the batteries. Circuit breakers protect the power distribution unit from dangerously high currents coming from the batteries. The *Power Storage* unit is divided into two major subsystems: the battery cabinet and the battery-load selection panel. The *Power Distribution* unit consists of two identical load banks. Each load bank is connected to the *Power Storage* unit and powers two DC loads and six AC loads. A complete description of the ADAPT system can be found in NASA 2006 [18].

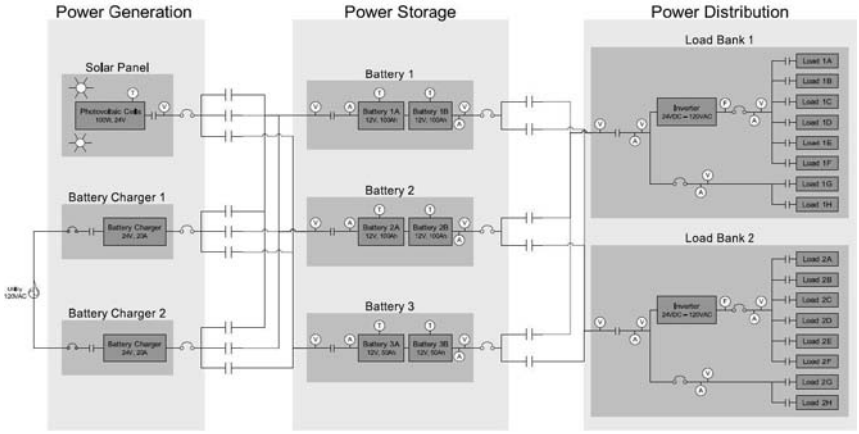

Fig. 6. The ADAPT testbed structure (picture reproduced from Poll et al. 2007 [19]).

The testbed is controlled by a number of relays and monitored by a large set of sensors. Consequently, it is possible to detect an injected fault and recover from it if the correct action is taken. To facilitate the execution of the experiments performed with the testbed, three operating roles have been defined by Poll et al. 2007 [19]: *user*, *antagonist* and *observer*. The user simulates an actual crewmember or pilot who operates and maintains the EPS with the help of a vehicle health management application. The antagonist injects faults into the system, either manually by physically acting on the system, or remotely by spoofing sensor values through a computer connected to the system. The malicious actions of the antagonist are not known to the user who is responsible of choosing a suitable recovery action. The

observer logs all data in the experiment and monitors how the user responds to the faults injected by the antagonist and therefore measures the effectiveness of the test article. The observer also acts as a safety officer of the experiment and can issue an emergency stop.

Several diagnostics systems such as HYDE (Narasimhan and Brownston [17]), FACT – Fault Adaptive Control Technology (Manders et al. 2006 [14]) from Vanderbilt University and TEAMS-RT – Testability Engineering and Maintenance System Real Time (Mathur et al. 1998 [15], Deb et al. 1998 [4]) have been reported to be integrated and tested on the ADAPT system.

At the 20th International Workshop on Principles of Diagnosis (DX-09) a diagnostics competition has been defined by NASA Ames and PARC with the goal to systematically evaluate different technologies and to produce comparable performance assessments for different diagnostics methods. The competition consisted of three different tracks. Each track defined a different diagnostic challenge problem. Two of the tracks were using real hardware test bed data based on the ADAPT system and the third track was based on ISCAS-85 and 74X-series benchmark combinatorial circuits proposed by Hansen et al. 1999 [7]. Those interested in the details, rules and set-up of the competition may wish to consult the description provided by Kurtoglu et al. 2009 [12] and Kurtoglu et al. 2009 [11].

5. EVALUTION OF THE MODEL-BASED DIAGNOSIS SYSTEM

The model-based diagnosis system has been tested in the context of the Second Diagnostics Competition DX-10. The ADAPT Tier 1 (ADAPT Lite) model from the DX Competition is depicted in Fig. 7 and contains a battery connected through a series of circuit breakers and relays to an inverter, and several loads consisting of a large fan, a DC resistor and AC resistor. The rotation speed of the fan is measured by a speed transmitter component. A series of four AC or DC voltage sensors and three current transmitters measure the voltage and current in different probing points of the circuit. The circuit breakers can be commanded externally to be closed or open and their position is monitored with the help of a position sensor connected to them. For each component type a separate library component has been created which has been instantiated in the model. The implementation details of the models are not relevant for the discussion in the paper. Let us just mention that for each component, the nominal behavior was modeled and augmented with the relevant failure modes. Some of the library components models are very detailed and the models were built by following the supplier specifications of the real component.

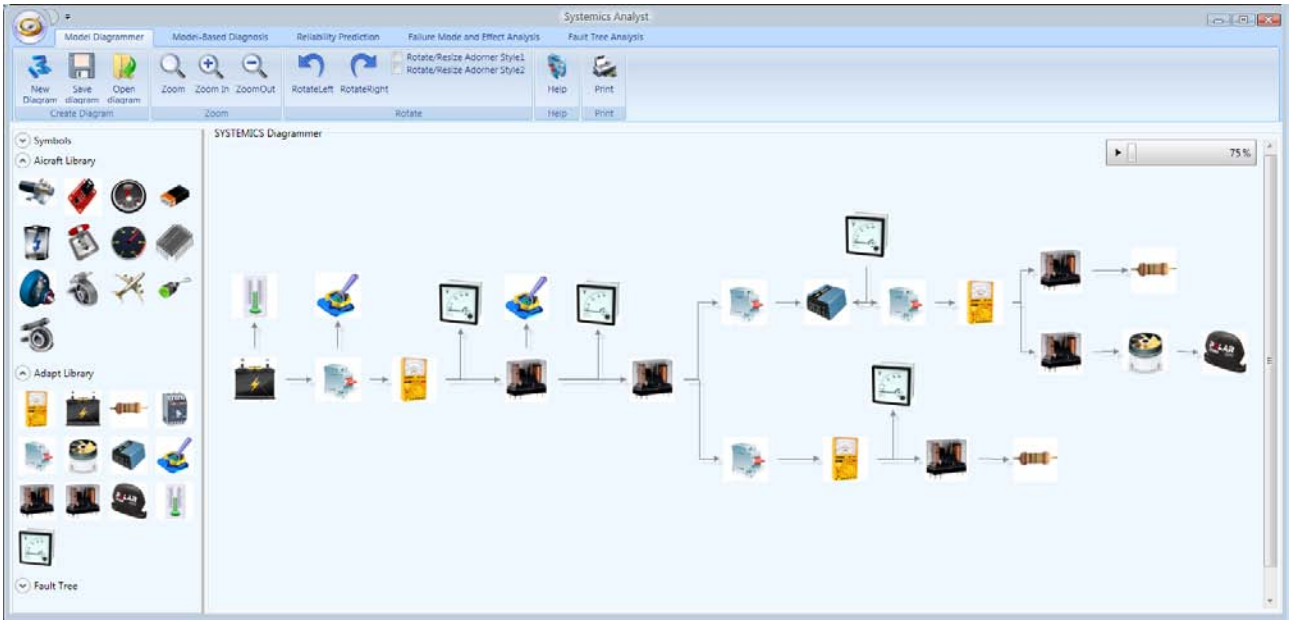

Fig. 7. The graphical view of ADAPT Lite modeled in Systemics Analyst.

Each sensor data frame was sent to the diagnostics engine via an application programming interface API and a diagnosis has been computed. The evaluation results performed on the available 39 experiments are given below:

DA: SystemicsC Diagnosis Algorithm
System: ADAPT-Lite
Mean Peak Memory Usage: 21733,8462 kb
Mean Time To Detect: 7312,697 ms
False Positives Rate: 0,0
Classification Errors: 1,0
False Negatives Rate: 0,0294

Mean CPU Time: 240036,8205 ms
Detection Accuracy: 0,9744
Recovery Cost: 325,0
Mean Time To Isolate: 11940,4242 ms

A formal description of the diagnostics metrics is described by Kurtoglu et al. 2009 [11]. It should be noted the high detection and isolation accuracy of the algorithm. For only one scenario our algorithm was not able to detect the injected fault.

6. SUMARY AND CONCLUSIONS

In this paper, we presented a prototype model-based diagnosis suitable for supporting the safety assessment process of avionics systems. The diagnostics capability of the environment has been demonstrated by applying several fault scenarios to the model for which diagnostics candidates have been generated. The model created for diagnostics is also suitable for simulation purposes. At the current stage of the implementation we are also able to automatically generate FMEAs directly from the model used for diagnostics. The model is considered to be the central repository of information from which different types of analysis that are part safety assessment process can be automatically generated. As a future work we intend to extend the environment to support the whole SAE ARP 4754 safety assessment certification process.

REFERENCES

[1] Airbus France, Eurocopter, Nexter Electronics, MBDA France, Thales Avionics, Thales Corporate Services, and Thales Underwater Systems. (2009) "FIDES Guide 2009. Reliability Methodology for Electronic Systems." 2009.

[2] Association Modelica (2007) "Modelica - A Unified Object-Oriented Language for Physical Systems Modeling - Language Specification Version 3.0," September, 2007.

[3] Deb Somnath, Sudipto Ghoshal, Amit Mathur, Roshan Shrestha, and Krishna R. Pattipati. (1998). "Multisignal Modeling for Diagnosis, FMECA, and Reliability." In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.* 11-14 October, 1998).

[4] Deb Somnath, A. Mathur, P.K. Willett, and K.R. Pattipati. (1998). "Decentralized real-time monitoring and diagnosis." In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.* (San Diego, CA, USA, 10-14 October, 1998). 3,

[5] Feldman Alexander, Jurryt Pietersma, and Arjan van Gemund. (2006). "All Roads Lead to Fault Diagnosis: Model-Based Reasoning with Lydia." In *Proceedings of the Eighteenth Belgium-Netherlands Conference on Artificial Intelligence (BNAIC-06).* (Namur, Belgium, October, 2006).

[6] Gould Eric. (2004). "Modeling it Both Ways - Hybrid Diagnostic Modeling and its Application to Hierarchical System Designs." In *Proceedings of the Annual Systems Readiness Technology Conference (Autotestcon 2004).* (San Antonio, Texas, September 20-23, 2004).

[7] Hansen M., H. Yalcin, and J. P. Hayes. (1999) "Unveiling the ISCAS-85 Benchmarks: A Case Study in Reverse Engineering." *IEEE Design and Test*, vol. 16: 3, pp. 72-80, 1999.

[8] Hayden Sandra C., Adam J. Sweet, and Scott E. Christa. (2004). "Livingstone Model-Based Diagnosis of Earth Observing One." In *Proceedings of the AIAA 1st Intelligent Systems Technical Conference.* (Chicago, Illinois, 20 - 22 September, 2004).

[9] Kleer Johan de and James Kurien. (2003). "Fundamentals of Model-Based Diagnosis." In *Proceedings of the IFAC SafeProcess.* (Washington USA, June, 2003).

[10] Kleer Johan de and Brian C. Williams. (1987) "Diagnosing Multiple Faults." *Artificial Intelligence*, vol. 32, pp. 97-130, 1987.

[11] Kurtoglu T., S. Narasimhan, S. Poll, D. Garcia, L. Kuhn, J. de Kleer, A. van Gemund, and A. Feldman. (2009). "First International Diagnosis Competition – DXC'09." In *Proceedings of the 20th International Workshop on Principles of Diagnosis (DX-09).* (Stockholm, Sweden, June, 2009).

[12] Kurtoglu T., S. Narasimhan, D. Garcia S. Poll, L. Kuhn, J. de Kleer, A. van Gemund, and A. Feldman. (2009). "Towards a Framework for Evaluating and Comparing Diagnosis Algorithms." In *Proceedings of the 20th International Workshop on Principles of Diagnosis (DX-09).* (Stockholm, Sweden, June, 2009).

[13] Lunde Karin, Rüdiger Lunde, and Burkhard Münker. (2006). "Model-Based Failure Analysis with RODON." In *Proceedings of the ECAI 2006 - 17th European Conference on Artificial Intelligence* (Riva del Garda, Italy, August 29 -- September 1, 2006).

[14] Manders Eric-J., Gautam Biswas, Nagabhushan Mahadevan, and Gabor Karsai. (2006). "Component-oriented modeling of hybrid dynamic systems using the Generic Modeling Environment." In *Proceedings of the Fourth*

Workshop on Model-Based Development of Computer-Based Systems and Third International Workshop on Model-Based Methodologies for Pervasive and Embedded Software. (Potsdam, Germany, March 30, 2006).

[15] Mathur A., S. Deb, and K.R. Pattipati. (1998). "Modeling and real-time diagnostics in TEAMS-RT." In *Proceedings of the American Control Conference*. 21-26 Jun, 1998). 3,

[16] Mauss Jakob, Volker May, and Mugur Tatar. (2000). "Towards model-based engineering: Failure analysis with MDS." In *Proceedings of the ECAI-2000 Workshop on Knowledge-Based Systems for Model-Based Engineering.* (Berlin, Germany, 2000).

[17] Narasimhan S. and L. Brownston. "HyDE- A General Framework for Stochastic and Hybrid Model-based Diagnosis." In *Proceedings of the 18th International Workshop on Principles of Diagnosis.* (Nashville, TN.

[18] NASA Ames Research Center (2006) "Advanced Diagnostics and Prognostics Testbed (ADAPT) System Description, Operations, and Safety Manual," February, 2006.

[19] Poll Scott, Ann Patterson-Hine, Joe Camisa, David Garcia, David Hall, Charles Lee, Ole J. Mengshoel, Christian Neukom, David Nishikawa, John Ossenfort, Adam Sweet, Serge Yentus, Indranil Roychoudhury, Matthew Daigle, Gautam Biswas, and Xenofon Koutsoukos. (2007). "Advanced Diagnostics and Prognostics Testbed." In *Proceedings of the International Workshop on Principles of Diagnosis (DX-07).* (Nashville, TN, May 2007, 2007).

[20] RTCA Inc. "DO-178B/ED-12B. Software Considerations in Airbone Systems and Equipment Certification."

[21] Sachenbacher Martin, Peter Struss, and Claes M. Carlén. (2000) "A prototype for model-based on board diagnosis of automotive systems." *AI Communications*, vol. 13: 2, pp. 83 - 97, 2000.

[22] SAE - System Integration Requirements Task Group. (1996) "ARP 4754. Certification Considerations for Highly-Integrated or Complex Aircraft Systems." 1996.

[23] SAE - The Engineering Society for Advancing Mobility Land Sea and Space International. (1996) "ARP4761 - Guidelines and Metods for Conducting the Safety Assesement Process on Civil Airbone Systems and Equipment." 1996.

[24] US Department of Defense, "*MIL-STD-217F. Military Handbook. Reliability Prediction of Electronic Equipment*," 1995.

[25] US Department of Defense, "*MIL-STD-882D. Standard Practice for System Safety*," 2000.