AN EXPERT SYSTEM FOR ONBOARD SYSTEMS PERFORMANCE ANALYSIS FOR SATELLITE LAUNCH VEHICLES

Dr. Sasikumar Punnekkat*, Vasantha Kumar KS#, Bindu Thomas* and Dr. Sudhakara Rao. K*

* Systems Reliability, Vikram Sarabhai Space Centre, Thiruvananthapuram-695022
# Department of Electronics, MIT Campus, Anna University, Chennai-600044

ABSTRACT

In satellite launch vehicles, any error in the onboard computer often leads to a mission failure and hence various verification and validation strategies are employed, to achieve the high reliability requirements. Typically for software validation, a large number of simulations are performed, with nominal condition, with disturbances in flight environments (including failure conditions) and with extreme dispersions in vehicle characteristics. Detailed analysis of onboard computer data available through telemetry during simulations is essential to establish that the performance of the onboard software is as expected.

We have developed an expert system to automate the analysis of simulation results. This is a rule-based system, where the rules depict the complex interdependencies among vehicle parameters. Its main feature is the ability to analyze any observed deviations in an autonomous way. This tool is being used to ensure the flight worthiness of onboard software and it has been found to accelerate the analysis and clearance processes.

1 INTRODUCTION

In satellite launch vehicles, a typical onboard configuration contains multiple computers in a cross strapped manner executing various onboard software such as navigation, guidance, digital autopilot, control, telemetry etc. In order to assess the performance of onboard software, various software verification and validation strategies such as code walkthrough, module testing and simulation studies are employed. Among these techniques, simulation runs play a major role and the onboard software used in a launch vehicle undergoes a large number of simulation runs to establish that the performance of the onboard software is as expected. An exhaustive analysis of the simulation test results is essential but cumbersome, because of

- Huge volume of data.
- Non availability of automated analysis tools
- Expertise spread over the entire organization
- Only a limited number of experts available to analyze the data at any particular time instant.
- Difficulty in analyzing complex interactions of parameters

As a result, automation is a must to acquire maximum information from the data generated by the simulation runs. Also it is essential to retain the invaluable knowledge of the experts, to take care of the scenario when they leave the organisation. These factors have motivated us to develop an expert system to analyze the performance of onboard system.

2 SYSTEM DETAILS

The Onboard Computer (OBC) system, which can be termed as the brain of the satellite launch vehicle, is perhaps the most critical element of all avionics systems deployed in the rocket. It consists of multiple processors like Navigation Processor (NP), Guidance Processor (GP), Control Processor (CP), which are interconnected. The OBC generates control commands of the vehicle during its flight by carrying out computations related to the navigation, guidance, autopilot and generation of mission critical event commands. In addition, the OBC is required to participate in pre-launch operations and to telemeter critical parameters to the ground during the flight as well as during ground tests.
The OBC architecture has been designed as two independent and identical computing systems with concurrent internal fault detection, performing identical functions to achieve the goal of single point fault tolerance. These are designated as Prime and Redundant (P & R) chains. Under totally error-free conditions during the flight, the prime chain controls the outputs. In case of failure of any computing element in Prime chain, its checking hardware disables its link and there is a switchover to the corresponding element in the redundant chain.

The main computing cycles for the OBC have been split into two specified tasks, major cycle (where the tasks of navigation and guidance are carried out) and minor cycle (where the tasks of vehicle control, digital filtering and vehicle sequencing are carried out). During the simulation runs in the test-beds, various parameters are telemetered, packed in a predefined telemetry format and sent to the checkout computer. To ensure correctness of onboard computer system, checking the software performance in the expected/nominal and abnormal simulated flight conditions is essential.

2.1 SIMULATION

Simulation is the process of modeling a physical system mathematically and making studies with the model by varying input conditions to analyse and improve the performance of the system. To validate the software, various flight conditions are simulated using purely mathematical models (known as Autonomous simulation) as well as in different test-beds such as Onboard computer In-Loop Simulation (OILS), Hardware in-Loop Simulation (HLS), Actuators in-Loop Simulation (ALS), etc., where the operating environments are modeled (vehicle model, environmental model, etc.) and various elements of the launch vehicle get exercised. The test cases are generated to simulate the flight environments with nominal conditions as well as with disturbances like propulsion dispersions, large separation disturbances, link failure cases, etc. Typical advantages of simulations are:

- The performance of onboard system can be evaluated at model level and changes can be made in the model to meet the requirements, prior to actual design and making of the system.
- The model can be tested to its extreme limits and the sensitivity of the system can be assessed.
- Development cost of the system can be reduced, as the faults can be detected and avoided
- Simulation paves a safer way for doing experiments, which are highly risky to do in reality.
- The tests can be performed for a large number of cases with simulation, which is not feasible with the actual system.

3 EXPERT SYSTEMS

An expert system is a computer program that simulates the reasoning of a human expert in a certain domain. To do this, it uses a knowledge base containing facts and heuristics and some inference procedure for utilizing its knowledge [1][2]. Expert systems typically contain a user interface as well as an expert interface (Fig. 1).

Fig. 1: A typical Expert System

The popular knowledge representation techniques are frame-based and rule-based methods.
3.1 FRAME-BASED EXPERT SYSTEM

Frame based knowledge representation uses a network of nodes connected by relations hierarchically. Nodes low in the hierarchy automatically inhibit the properties of higher-level nodes. The employment of frames represents a robust way to present the knowledge. A frame contains an object plus slots for all information related to the object. The contents of such slots are typically the attributes and attribute values of the particular object. Additionally, slots may contain default values, pointers to other frames, and associated sets of rules or procedures. Though frames are versatile, their usage could rather become complex which is their primary drawback, ironically caused by the robustness of such a mode of representation [3].

3.2 RULE-BASED EXPERT SYSTEM

In a rule-based expert system, the knowledge is encoded in a declarative form, which comprises of a set of rules of the form ‘situation → action’ [3]. These rules are called production rules or IF-THEN rules. In a rule-based expert system, the domain knowledge is represented as sets of rules, which are checked against a collection of facts or knowledge about the current situation. Facts are pieces of information that can be used by the expert system and refer to specific cases.

When the IF portion of a rule is satisfied by the facts, the action specified by the THEN portion is performed. When this happens, the rule is said to fire or execute.

3.3 KNOWLEDGE BASE

The facts and rules that represent the modeling constitute the knowledge base. To be useful, the knowledge must be accurate, presented at the right level for encoding, complete in the sense that all essential facts and rules are included, and must be free of inconsistencies. The knowledge base contains the domain-specific knowledge required to solve the problem. Knowledge structures are used to store knowledge and reason with it, just as data structures are used to store data and work on it. Since there can be many inputs (often termed as ‘evidence’) and outputs (referred as ‘conclusions’), as well as hundreds or even thousands of rules in the knowledge base, it is significant/mandatory to have a mechanism to search through the rules and find the appropriate output. The algorithm for performing this is called an inference engine. The knowledge used by an expert system needs to be represented and employed in a form so that it can be used for reasoning by an automated inference algorithm. This is in contrast to most computer programs that work with data.

3.4 THE INFESSION ENGINE

The inference engine, which serves, as the inference and control mechanism is an essential part of the expert system and is a major factor in determining its effectiveness and efficiency. Inference is the process of drawing a conclusion (either intermediate or final) for specific set of facts for a given situation by means of a set of rules. Facts by themselves cannot be used for reasoning. We need to associate the facts along with the rules to reason and drive another fact, by searching for a solution. The two fundamental search strategies employed by an expert system are:

- **Forward chaining** is an inference method where rules are matched against facts to establish new facts. Forward chaining proceeds from data to conclusion, and is said to be data driven. If one has a few premises and many conclusions, then forward chaining is the best suitable strategy.
- **Backward chaining** is an inference method where the system starts with a tentative conclusion (i.e., what it wants to prove) and proceeds backwards to the premises to determine if the data supports the conclusion. This is a goal driven approach and is best suited, if one has many premises and relatively few conclusions.

Both approaches will lead to a conclusion, but their search efficiency - depends on the nature of the inference network associated with the problem. Popular languages used in developing expert systems are LISt Programming language (LISP) or PROgramming language for LOGic (PROLOG). Some of
the well-known packages used for building expert systems are OPS5, EXPERT, INSIGHT, KNOWLEDGE CRAFT, TEKNOWLEDGE, etc [4].

4 SYSTEM REQUIREMENTS

Our system requirements are four-fold, viz., telemetry data extraction, telemetry data analysis, expert system for decision making and expert system management. Typical requirements are:

- Extraction of parameter values for all test cases from the OBC telemetry data files.
- Support for text and graphics modes of analysis outputs
- Analysis and comparison of prime & redundant processor data
- Analysis and comparison between data obtained from various simulation test-beds
- Comparison of data with varying sampling rates
- Expert analysis of any parameter, and in case of anomaly, analysis of the cause based on rules
- Provision for user interaction throughout the analysis.
- Facility for experts to gather, display and revise rules of the knowledge base
- Protection to avoid unwanted / unauthorized access and updating of knowledge base.

In addition, typical usability requirements such as configurable file locations, access across LAN, user friendliness, detailed error reporting and dynamic memory management are to be taken care of.

5 DESIGN AND IMPLEMENTATION DETAILS

A rule-based expert system has been developed to automate telemetry data analysis. Choice of rule-based knowledge representation has been made for the following reasons

- Rules represent knowledge in a natural mode. Consequently, the time required to learn, how to develop rule bases is minimal.
- Rules are transparent as compared to frames or other modes of representation
- Rule bases can be easily modified. In particular, addition, deletion, and revisions to rule bases are relatively straightforward processes.

We did not use standard packages [5] for developing expert system due to their prohibitive costs and stand-alone mode of operation. One of the major requirements was the necessity to integrate the expert system with existing utilities for acquisition and conversion of the binary telemetry data and hence the integrated analysis framework including the expert system was developed using Visual C++ in Windows. For graphical analysis, the EASY PLOT package was integrated with the software.

The system consists of modules for parameter extraction, parameter analysis, expert system management and expert analysis. We now describe the functions of expert system related modules.

5.1 EXPERT SYSTEM

Expert system management module allows the expert to add, display, modify or delete rules. Each rule is of the form ‘if antecedent then consequent’. Existing rule types are as follows:

**Type 1**  If Parameter-X in OBC-A is outside the threshold limits (Max or Min) continuously for N cycles then check Parameter-Y in OBC-B.

**Type 2**  If difference between OILS and AUTONOMOUS values of Parameter-X in OBC-A exceeds the Max value for N cycles then check Parameter-Y in OBC-B.

**Type 3**  If Parameter-X in OBC-A oscillates continuously for N cycles then check Parameter-Y in OBC-B.

OBC-A/B could be any of the onboard computers such as GP, NP or CP and Parameter-X/Y could be any of the vehicle parameters (obviously X and Y cannot be the same parameter). This module also has the facility of adding new rule type according to the nature of the anomaly.
The main features of the expert management module are:

- User information is processed using menus to frame the rule structure.
- Parameters are listed dynamically corresponding to the OBC package
- User can add as many rules he wishes to the existing knowledge base.
- Confirmation from the user is sought before processing (add, modify or delete) a rule.

5.2 ANALYSIS STEPS

Our aim is to automate the expert data analysis by an ‘anomaly-driven’ approach. The steps involved in the design of this analysis system are described in following subsections.

5.2.1 Classification of subsystem parameters

Each subsystem, which gets tested in the simulation run like control system, guidance system, sequencing system, is analysed in detail and the list of parameters which enable the user to arrive at a conclusion about the performance of the system in that particular run, is identified. For e.g., to analyse the performance of Control subsystem, the relevant parameters identified are control commands in pitch, yaw and roll axes, the body rates and the error commands.

5.2.2 Extraction of parameters

In the onboard system, there exist tasks, which are performed in two periodicities, i.e., minor cycle (20 ms) or major cycle (500 ms). Each processor telemeters a frame of data in every minor cycle with a timestamp and the minor/major cycle parameters should be appropriately extracted. For example, as control function is exercised every 20 ms, its parameters are telemetered in every 20 ms. But guidance task is a major cycle task and hence the guidance parameters are telemetered every major cycle. Based on the type of onboard processor and the type of parameter, the support for extracting all parameters is implemented in the system, utilizing object-oriented methodology.

5.2.3 Preliminary analysis of a run

(i) Extraction of the orbital parameters

The orbital parameters i.e., apogee, perigee and inclination, which give an overall feedback about the performance of the onboard system for the particular simulation run are extracted and checked against the mission specifications.

(ii) Extraction of various flags – event flags and error flags

The onboard system is designed in such a way that the general health of various subsystems, software performance and execution state of the system are available through a set of status bytes, known as ‘flags’. Each bit of these bytes can indicate vital information about the system status like

(i) Accelerometer failure
(ii) Inter-processor link failure
(iii) Overflow detected in software computation
(iv) Occurrence of an important vehicle sequencing event

It is automatically checked whether all correlated flags are consistently being set. For e.g., if an accelerometer failure bit is set in a flag, the reason of failure will be indicated in another flag. Based on the analysis of the flags, an overall conclusion can be drawn on the particular simulation run.

(iii) Quick look analysis

After the preliminary analysis a quick look analysis report is available to the user and he can decide further steps to be followed to analyse the anomalies, if any, reported.
5.2.4 Invoking the expert analysis system

The user can select any parameter to be analysed in detail, from the list of parameters. The system first extracts the parameter and searches the rule base to find whether the parameter forms part of the antecedent. Suppose user has selected the option to analyse the control commands in a simulation run. The rule base has a Type –1 rule, which states that

“If there is a sudden transition in a control command output, then check the corresponding body rate during a defined neighborhood of the time instant”

To check whether the rule should be fired for the test data being analyzed, procedures are invoked with the extracted control command as input. As output of the procedure, the time duration of a sudden transition if any in the control command and the plot of the parameter at that duration is displayed on the screen.

The expert system then proceeds to analyse the cause of the anomaly by checking whether the conclusion as depicted in the rule is satisfied during the period. In a similar way all applicable rules are checked and graphs/conclusions provided to user. Thus the system guides the user in an interactive way to conclude as to what caused the selected parameter to deviate from its expected behavior.

For type-2 rules, the parameters are compared with the results of a high level equivalent program. There exists a difference in the periodicities of the parameters in these two simulations and so linear interpolation technique is used to bring the values in the same time scale. Ideally the values of the parameters should match within a specified tolerance in the two modes of implementation. Here, the rules are framed as in the following example:

“If the difference between control commands in GP and that in autonomous is more than 0.5 volts for more than 2 seconds, then check whether there is difference in body rates between GP and autonomous”.

With type-3 rules, special utilities are called to detect oscillations in any given parameter and check further for the cause of the oscillations based on related rules from the rule base.

5.2.5 General features of the system

- Parameter file, which contains the parameter details such as scale factor, unit, starting byte position etc., are in text format, so that modifications can be done easily.
- All the modules have menu bar. So one can select any utility from any module.
- In case of error in any part, appropriate error message boxes will be displayed
- Before decisions, confirmation from the user is solicited for protection.
- As most of the required data field has to be selected only through combo boxes, chance of entering parameters wrongly is ruled out.

5.2.6 Sample result/plots

The following are a sample result and plot given by expert system tool:

![Expert System Analyse](image.png)

Fig 2. System message upon analysis of control output dapcl
A “proof of concept” prototype model of the expert analysis system has been successfully developed and demonstrated. The system is menu-driven, user-friendly and features a customizable rule-management module. In case of parameter extraction module the results showed very good match with the outputs of existing in-house developed utilities except for a few precision issues observed in some major cycle parameters, which could be attributed to differences in declarations of internal variables of the programs. The analysis will continue till all the rules in the knowledge base are checked throughout the flight regime. Overall, the results are encouraging and efforts are on to implement this system in a full-fledged form for regular use.

Following are some of the improvements being implemented and future extensions planned:
- Extending the system to different onboard computers and all types of launch vehicles.
- Automated verification of simulation objectives by incorporating test case specifications
- Provision to handle multiple parameters or all the parameters at one stretch and providing a summary of results with trace
- Optimizations to take care of real systems with large number of rules.
- Adding training facility and keeping typical scenarios automatically as examples.

REFERENCES

ACKNOWLEDGEMENTS
The authors wish to acknowledge the technical suggestions and help rendered by Smt. Uma Sankari SS, Shri Vijay Srinivasan PS, Shri Suresh V Mathew, Dr. Sukumaran Nair CG, and Dr. Vikraman Nair R. The authors are also grateful to Shri G. Madhavan Nair, Director, VSSC for the encouragement given to this expert system development activity.