

---

# **PTDS: Space Shuttle Main Engine Post Test Diagnostic Expert System for Turbopump Condition Monitoring**

**Pamela Surko**

Science Applications International Corp.

**June F. Zakrajsek**

NASA Lewis Research Center



**SAE** *The Engineering Society  
For Advancing Mobility  
Land Sea Air and Space®*  
**INTERNATIONAL**

**Aerotech '92  
Anaheim, California  
October 5-8, 1992**

---

**400 COMMONWEALTH DRIVE, WARRENDALE, PA 15096-0001 U.S.A.**

# PTDS: Space Shuttle Main Engine Post Test Diagnostic Expert System for Turbopump Condition Monitoring

Pamela Surko

Science Applications International Corp.

June F. Zakrajsek

NASA Lewis Research Center

## ABSTRACT

A health monitoring expert system software architecture has been developed to support condition-based health monitoring of rocket engines. Its first application is in the diagnosis decisions relating to the health of the high pressure oxidizer turbopump of the Space Shuttle Main Engine. The post test diagnostic system runs offline, using as input the data recorded from hundreds of sensors, each running typically at rates of 25, 50, or .1 Hz. The system is invoked after a test has been completed, and produces suggestions, analysis, and an organized graphical presentation of the data with important effects highlighted. The overall expert system architecture has been developed and documented so that expert modules analyzing other line replaceable units may easily be added. The architecture emphasizes modularity, reusability, and open system interfaces so that it may be used to analyze other engines as well. In its first application, the expert system has identified drifting sensors, anomalous shaft motion and preburner pump bistability.

## INTRODUCTION

The Post Test Diagnostic System (PTDS)\* aids engineers who are responsible for detecting and diagnosing engine anomalies from sensor data by providing a consistent, fast first-pass data analysis. The analytical methods used by PTDS are modeled after these engineers' analysis strategies\*\*. The modular architecture has both procedural and nonprocedural knowledge-based components. This combination allows for the inclusion of conventional algorithms as well as heuristic expert information. Currently this architecture has been implemented with the expert modules that analyze the High Pressure Oxidizer Tur-

bopump (HPOTP).

The safe and reliable operation of a rocket engine depends on the proper functioning of each individual component of the engine. To ensure successful engine operation many engineers and technicians test, troubleshoot, and where possible monitor critical components during operation. The Civil Space Technology Initiative's Earth-To-Orbit program has funded the development of several post-test, post-flight diagnostic techniques which are being incorporated into PTDS. The near-term benefit of this system is to provide the engineers with an expedient means of reducing and interpreting the large amounts of data which reduce the time and manpower required after engine tests and flights.

All data being analyzed is time series data. Interaction between cooperating expert modules must incorporate time-dependence of features. A large system such as PTDS can grow too complex if care is not taken to provide only functionality that is essential. Therefore, a simple representation of time dependence that still has sufficient power for this domain was chosen. The system allows multiple snapshots of the domain at different times, explicitly handling points in time (snapshot times) and time intervals. It also handles specific time dependence over an interval of an individual sensor trace. The following sections describe the designs of the session manager, database schema, expert modules, reasoning strategies, storage of expert system results, the user interface, and implementation details.

## ARCHITECTURE

**OVERVIEW** - The Post Test Diagnostic System operates off line and with minimal human assistance. The system requires notification that a new test data set has arrived, and the input of several unit numbers of the engine Line Replaceable Units (LRU) being tested. The PTDS analyzes the data and prepares a results table for inspection by the data analysts. This process is typically conducted overnight and is ready for the data analyst in the morning.

\* Surko, Pamela, Reusable Rocket Engine Turbopump Health Management System, contract NAS3-25882.

\*\* Zakrajsek, June F., The Development of a Post-Test Diagnostic System for Rocket Engines, AIAA-91-2528. June, 1991.

The data analysis process is implemented using the architecture shown in Figure 1. The controlling module is the "session manager." This UNIX process loads the numerical data from unstructured binary files into the relational database management system (RDBMS), runs the feature extraction routines, and coordinates the expert modules. The expert modules reason using the features, static knowledge such as limits and expected noise levels contained in tables, and the knowledge contained in the rules. The user interface is a separate process, which queries the results, features, and raw numerical data tables, to provide an intelligent display of the results.

**SESSION MANAGER** - The session manager controls the flow of data and the expert modules that analyze the data. It manages the execution of the expert modules by inspecting a resource table maintained in the RDBMS, to determine whether the prerequisites for running a module exist, and if so, then invoking it as an independent UNIX process. This allows individual expert modules to be written in different languages, and gives individual expert module developers maximum flexibility.

A very simple mechanism for allowing modules to interact with each other has been implemented. A module can request information from other modules by writing to an RDBMS table which functions as the PTDS blackboard. If a request is written to the blackboard, the session manager reinvokes the requested module, then reinvokes the requester. All modules are responsible for inspecting the blackboard when they are invoked. To avoid deadlock, a module is required to be reinvoked before it can read the blackboard, and to avoid looping each module can not be invoked more than twice, an arbitrary but satisfactory limit.

**DATABASE SCHEMA** - The test data is managed in an *ssme\_data* Ingres database, even though the official archiving of the numerical data is done elsewhere, using binary unstructured files. The RDBMS implementation strategy was chosen to ensure a robust system and to ease the porting of the system to the analysis of other engines. The features of data security, user privileges, checkpointing, and backup were provided by the commercial software and therefore could be eliminated from the custom software development.

The tables chosen for storing the test data is shown in Figure 2. The table *test\_info* contains one record per test, and has all the data that appears once per test, such as test number, date and time of day of the test, LRU unit numbers being tested, and the length of the test. The table *pid\_info* contains one record per sensor, per test. Its fields store the information appearing once per sensor for each test. A record contains the parameter identification number (PID), a description of what the PID is measuring on this test, the engineering units, the sensor type, and the sampling rate, typically 25, 50, or .1 Hz. The final table shown contains the numerical data for all sensors, for a particular

test.

The table structure was chosen to optimize the speed of two types of queries: more important, the queries done by the user interface to retrieve data for display to the user; and second, the types of queries done by the feature extractor for its work. The feature extractor queries the database more often than does the user interface. However, this module operates in batch mode, typically before the data analysts arrive, therefore subsecond response times are not necessary. The queries required by the user interface retrieve and graphically display data while the analyst waits, therefore display time is crucial.

The time required to load the test data ranges from 20 minutes to more than an hour, depending on the length of the test firing. This loading time is slow, but since it is done overnight in batch mode, decreasing the load time was not necessary. A typical engine test totals 20-60 Mbytes, depending on the length of the test. Currently the system has 10-20 recent test firings resident in the database. The number of tests available on-line is limited by the space available on a 1 Gbyte file system.

The RDBMS also manages feature tables, which store the intermediate results from curve fitting and other algorithmic analysis done on the numerical time series data, and results tables, which contain the observations to be displayed to the user, and instructions for formatting the graphical displays to support those observations. The session manager has its own small database of tables containing instructions for managing the feature extraction process, and tables of prerequisites for each process, allowing the session manager to be written without knowledge embedded into it of what order the various modules should be invoked, or what files or tables need to be present in order for them to function correctly. Since modules may be added throughout the lifetime of PTDS, the session manager must be as generic as possible.

**EXPERT MODULES** - One group of expert modules has currently been completed. These modules analyse the seals, balance piston, and preburner pump bistability of the High Pressure Oxidizer TurboPump (HPOTP). The expert knowledge for the analyses is resident in three places: first, the list of which features to search for in which sensor traces; second, the tables storing the knowledge which does not change from test to test, used by the rules to define limits, allowed variability, expected noise levels and the like; and third, the knowledge embedded in the rules themselves\*. All numerical static knowledge is stored in RDBMS tables rather than being hardcoded into rules or algorithmic code, to allow for ease of expansion and transfer. It is especially valuable to plan for maintainability

\* Surko, Pamela, Task 1: Expansion of Existing Health Monitoring Logic. Reusable Rocket Engine Turbopump Health Management System, contract NAS3-25882.

in expert system development, since experts who have previously judged effects only by eye, using available hard copy graphs, may wish to try several values for bounding values.

**REASONING STRATEGIES** - The system reasons about whether two engine effects, as manifested in different sensors, may be related to a common cause or to each other, based on the time behavior of each occurrence. Also, PTDS reasons about the internal time structure of a single sensor signal by characterizing signals as "erratic", "spike", "peak", or "level\_shift". Currently time dependence is implemented with one data structure (class), the *feature*, whose time-management properties are often simply start time and duration. Some features, such as asymmetric peaks in a time plot, have more complex time dependence, and the system carries the time of the maximum, and of both full-width-half-max points. Features are generated for all events of interest in the raw sensor data. Events are time-compared to other events using only the relations *before*, *simultaneous\_with*, *during*, and *overlapping*.

Nearly all reasoning done by expert modules can be done with an appropriate set of features, rather than the full numerical data set. For efficiency's sake, the expert system does not reason directly about the individual data points, but about the features the experts have identified as important in the data. First, individual sensor traces are examined for expected behaviors, and a small set of useful curve-fitting routines are run to identify events in the time series data that are recorded as features. Only a small number of features are necessary to characterize most of the behaviors the expert system must analyze, such as "peak", "spike", "flat", "level shift", "different than" (comparing two traces) and "erratic behavior". The features are written to a database table and these, rather than raw sensor numerical data, are used as the symbols about which the expert modules reason. The expert module can then reason about features.

Two analysis strategies used by experts are used in the expert system. The current test is compared to results from a previous test to note unexplained differences, and the current test is also examined for evidence of a specified set of problems.

In searching for unexpected areas of change in the behavior of the turbopump between the current test and a previous one, one would hope to compare the current test to the previous test running the same turbopump. This would be relatively simple, were it not for the fact that rarely are the operating conditions identical for two tests. The thrust profiles or the tank pressurization profiles may differ. The presence of a different low pressure turbopump, a different fuel turbopump or a different fuel/LOX mixture ratio will affect the operation of the HPOTP. In comparing two tests, each test is segmented into periods of constant

thrust, by extracting level-change features on the thrust sensors, to identify the intervals at which the thrust was changing, and then identifying the time intervals between those thrust changes as periods of constant thrust. Each period of constant thrust is tagged with the actual value of thrust during that period, so that when comparing traces that should be the same within tolerances, comparisons are only done during periods of relative engine equilibrium, under similar thrust conditions. Thrust is the main driver of expected differences between tests. Rules have been designed but not yet implemented, to manage differences in tank pressurization as well.

An important piece of knowledge in doing test-to-test comparisons is the knowledge of which previous test should be used in the comparison. Each expert module makes its own determination of which previous test to use in current-previous comparisons. LRUs are swapped fairly often. For example, if the most recent previous test used the same HPOTP as the current test, but a different fuel turbopump, the system might choose that test for HPOTP comparison analysis, but choose the most recent test using the same fuel turbopump for a fuel turbopump analysis. In HPOTP analysis, the decision is made based on half a dozen criteria, with differing priorities. The system first searches for the most recent previous test on the same test stand, with the same engine, and the same HPOTP. Quite a number of rules handle the cases where not all the criteria are met. In order for the system to make best use of previous test data in doing these test-to-test comparisons, tests are analyzed in chronological order.

PTDS discards expected differences, and reports unexpected differences. This strategy reports unexplained new behavior of a pump without requiring knowledge of what caused that difference. Problems never before observed are detected, even though specific diagnostic rules for the anomaly are not present.

In the second reasoning strategy, using data from the current test only, several types of reasoning are done. One type exploits the limited redundancy available in the sensor data, by comparing both sensors to a previous test, if they differ, and assuming that if only one disagrees, that the physical quantity being measured agrees with the previous test and that one disagreeing sensor is faulty. If a parameter is sensed by only one sensor, then the dual hypotheses of actual physical change and sensor failure are made, unless other constraining evidence is available. The majority of the expert system rules treat the diagnosis of particular failure modes. For example, the balance piston module searches for various patterns and correlations in the traces from the two pressure sensors monitoring the pressures providing restoring forces for the pump impeller. Spikes or level shifts in both pressures mean possible anomalies in the axial position of the impeller shaft. Spikes or level shifts in one pressure only imply different problems, and

the relative sign of the spikes or level shifts give further clues as to what anomalies occurred. The system reports the unexpected features it detected, and groups them where possible under a common root cause ("possible damage to a high pressure orifice").

**STORING EXPERT SYSTEM RESULTS** - For each anomaly, a record is written by the expert module detecting it to the *results* table of the *ssme\_data* database. This table is accessed by the user interface when a user logs in to view the results of a test.

Using the Ingres table as a cache for results allows expert modules to be added to the system without requiring extensive changes to the user interface. The features for a test are also written to the features table by the feature extractor module, which runs before the expert modules.

When old tests are deleted from the database to release space, the features and results for that test are not deleted. Only the large numerical data table is deleted. This allows PTDS to return to older tests if they are appropriate for previous-test comparison.

**USER INTERFACE** - The point-and-click color user interface allows the data analyst to view PTDS observations and supporting graphs. When the user invokes the system, the first window lists the tests residing in the system, and allows the user to choose the desired tests. An active engine diagram is offered, with engine LRU's highlighted if the expert module analyzing that LRU has detected anomalies. The user selects an LRU, highlighted or not, to see a display of the detailed schematic of that LRU annotated with PIDs, and the observations associated with the LRU. If an anomaly is present in any PID, its label on the schematic is highlighted in color. By choosing a PID label or an anomaly line, the analyst displays the related time series data graphs. PTDS scales the data to highlight the important time segment, as well as displaying the supporting graphs often consulted by experts that show general engine conditions.

**IMPLEMENTATION DETAILS** - The PTDS system is resident on a Sun SparcStation 1, with an external SCSI hard disk drive of 1 Gbyte. Three commercial software packages were used: Nexpert Object from Neuron Data, the Ingres relational database management system, and the PV-Wave display package from Precision Visuals. The system runs under UNIX.

The user interface was written using the X11R4 X-windowing package, in Motif-compliant C code. The feature extraction functionality was written largely in C, using published curve fitting algorithms\*. Code was documented for reusability. Slightly more than 1000 Nexpert rules cover feature filtering to eliminate uninteresting

features; diagnosis of HPOTP balance piston problems; HPOTP preburner pump bistability; and diagnosis of HPOTP seal problems.

#### CONCLUDING REMARKS

The Post Test Diagnostic System has been applied to the High Pressure Oxidizer Turbopump of the Space Shuttle Main Engine. The system has been in operation at Marshall Space Flight Center for several months, and new tests are being run through the system. Limit table entries have been chosen, and the system is producing observations. There are currently several incorrect diagnoses that show up regularly, and the developers have already designed a more sophisticated rule set that will eliminate the problem. Even though the Post Test Diagnostic System is not completely tested and the High Pressure Oxidizer TurboPump knowledge base is not yet complete, it has proven its potential usefulness. The system has successfully identified drifting sensors, and anomalous shaft motion. It also detected a subtle case of preburner pump bistability on a nonflight turbopump that was missed initially by human analysts. It should be noted that analysts are not required to report bistability on test turbopumps.

Expert system technology has come of age, and can provide valuable diagnostic help to time-pressed analysts, by automating tedious aspects of their job. The Post Test Diagnostic System has been developed using a modular distributed architecture. It combines both procedural and heuristic code to detect and diagnose anomalies present in test data.

\* Press, William H., Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes in C; the Art of Scientific Computing*, Cambridge University Press, 1988.

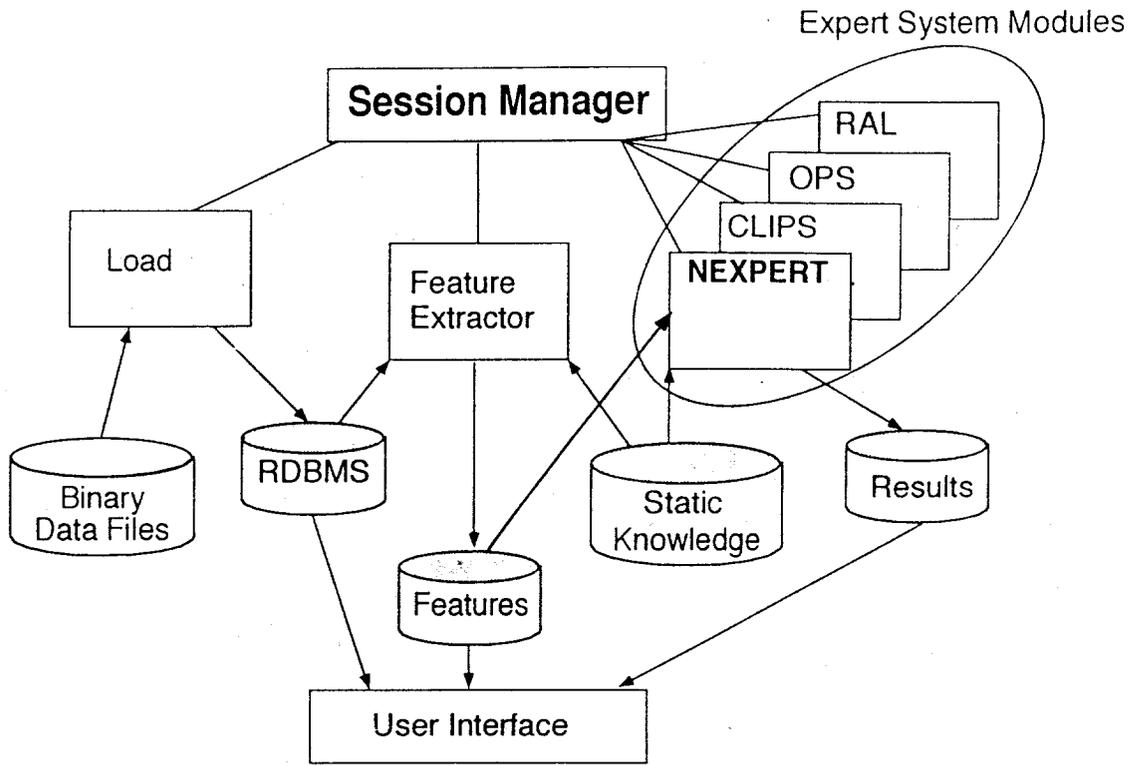


Figure 1: Architecture Overview

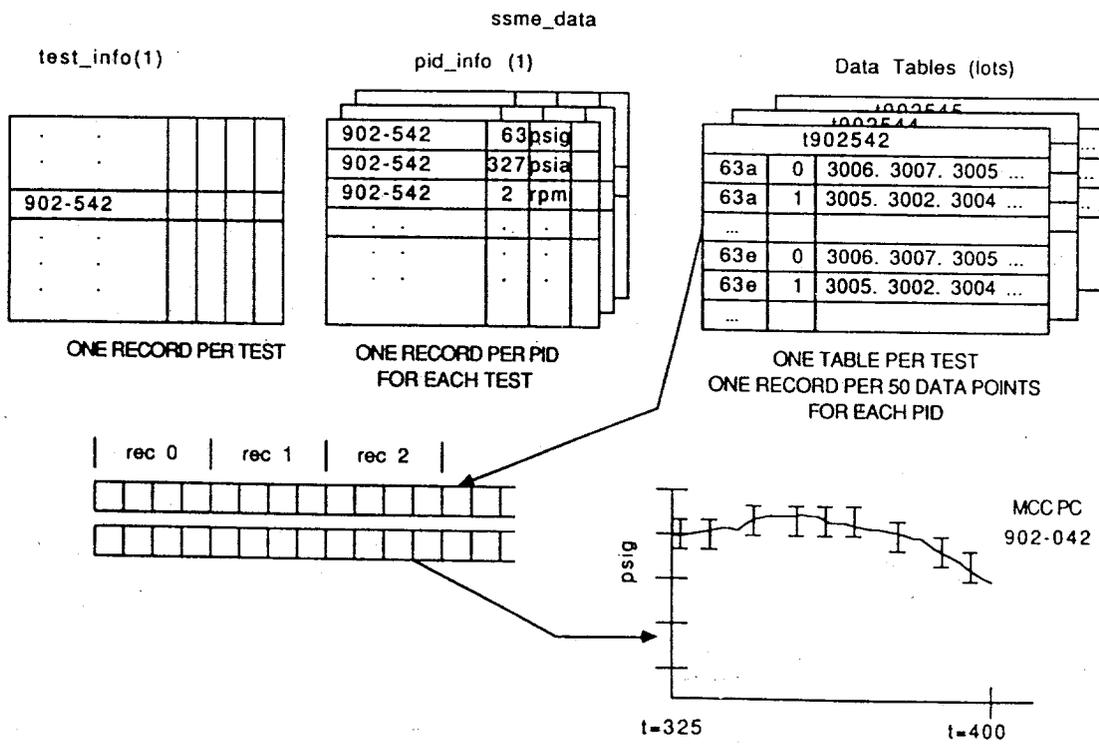


Figure 2: Database Table Design

The appearance of the ISSN code at the bottom of this page indicates SAE's consent that copies of the paper may be made for personal or internal use of specific clients. This consent is given on the condition, however, that the copier pay a \$5.00 per article copy fee through the Copyright Clearance Center, Inc. Operations Center, 27 Congress St., Salem, MA 01970 for copying beyond that permitted by Sections 107 or 108 of the U.S. Copyright Law. This consent does not extend to other kinds of copying such as copying for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale.

SAE routinely stocks printed papers for a period of three years following date of publication. Direct your orders to SAE Customer Service Department.

To obtain quantity reprint rates, permission to reprint a technical paper or permission to use copyrighted SAE publications in other works, contact the SAE Publications Group.



*All SAE papers, standards, and selected books are abstracted and indexed in the SAE Global Mobility Database.*

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

**ISSN 0148-7191**

**Copyright 1992 Society of Automotive Engineers, Inc.**

Positions and opinions advanced in this paper are those of the author(s) and not necessarily those of SAE. The author is solely responsible for the content of the paper. A process is available by which discussions will be printed with the paper if it is published in SAE transactions. For permission to publish this paper in full or in part, contact the SAE Publications Division.

Persons wishing to submit papers to be considered for presentation or publication through SAE should send the manuscript or a 300 word abstract of a proposed manuscript to: Secretary, Engineering Activity Board, SAE.

**Printed in USA**