Vehicle-Level Reasoning Systems: Integrating System-Wide Data to Estimate the Instantaneous Health State

Ashok N. Srivastava, Ph.D., NASA Ames Research Center
Dinkar Mylaraswamy, Ph.D., Honeywell
Robert Mah, Ph.D., NASA Ames Research Center
Eric G. Cooper, NASA Langley Research Center

1 Introduction

One of the primary goals of Integrated Vehicle Health Management (IVHM) is to detect, diagnose, predict, and mitigate adverse events during the flight of an aircraft, regardless of the subsystem(s) from which the adverse event arises. To properly address this problem, it is critical to develop technologies that can integrate large, heterogeneous (meaning that they contain both continuous and discrete signals), asynchronous data streams from multiple subsystems in order to detect a potential adverse event, diagnose its cause, predict the effect of that event on the remaining useful life of the vehicle, and then take appropriate steps to mitigate the event if warranted. These data streams may have highly non-Gaussian distributions and can also contain discrete signals such as caution and warning messages which exhibit non-stationary and obey arbitrary noise models. At the aircraft level, a Vehicle-Level Reasoning System (VLRS) can be developed to provide aircraft with at least two significant capabilities: improvement of aircraft safety due to enhanced monitoring and reasoning about the aircraft's health state, and also potential cost savings through Condition Based Maintenance (CBM). Along with the achieving the benefits of CBM, an important challenge facing aviation safety today is safeguarding against system- and component-level failures and malfunctions.

A VLRS can take advantage of component, subsystem, and vehicle-level models which would represent connectivity and potential causal chains of failure. Moreover, physics-based models of damage propagation for certain subsystems (such as the airframe or actuators) may be appropriate for inclusion in the model. Finally, data-driven methods to characterize interactions between components, subsystems, and systems may be appropriate for the design. The architecture of a VLRS can span a wide range of aircraft subsystems such as airframe, propulsion, avionics, and software system.

Vehicle level reasoning takes into account health management information from all levels – component, subsystem, system, and fleet-wide. The reasoning at each level, which has its own requirements in terms of timing, processing, and communications, is aimed at disambiguating any conflicting information and improving situational awareness. The level of Verification and Validation (V&V) scrutiny of the VLRS itself will be determined by the severity of the aircraft-level hazards associated with it. Advanced VLRS concepts such as the integration of data from airborne and ground systems, use of active query for fault isolation, interaction with the flight crew, and characterization of component, subsystem, and system interactions through data-driven methods, will likely pose new and significant verification, validation, and certification challenges.

This chapter begins with a background on reasoning systems that were first developed for space applications and then discusses the concept of a vehicle reasoning system with aircraft applications with a real-world example and case study. We then discuss some critical issues regarding the verification and validation of such a system and then summarize with a section on conclusions and future work.

2 Background

The need for autonomous spacecraft and rovers has been a driving force for the development of vehicle-level reasoning technologies. These technologies are applicable for both unmanned aerial vehicle and next generation commercial aircraft. This section describes a few examples that demonstrate the feasibility and benefits of reasoning systems for space applications.

In 2004, NASA uploaded Livingstone Version 2 (LV2) software to the EO-1 satellite to test its ability to find and analyze errors in the spacecraft's systems [1]. Normally such troubleshooting is performed on the ground. Tests were conducted to automatically detect and diagnose simulated failures in the satellite's instruments and systems. Livingstone provides the opportunity to recover from errors to protect these assets, and continue to achieve mission goals. On this mission, LV2 also monitors another software application that controls EO-1 to autonomously run its imaging system. If EO-1 does not respond properly to the software control, LV2 detects the error, makes a diagnosis, and sends its analysis to mission control. LV2 compares a model of how the spacecraft's systems and software should perform to actual performance. If the spacecraft's behavior differs from the model, then the LV2 reasoner searches for the root cause and gives mission controllers suggestions of what may have gone wrong.

Another example of a space-related reasoning system is Remote Agent. This system enables autonomous planning and execution of many tasks onboard the spacecraft [2]. With this capability, only general directions are commanded from ground controllers on earth. This allows faster response by the spacecraft to in-flight situations since ground controller intervention is limited due to communication delay. As an example, autonomy capability is needed to safely maneuver a spacecraft in a hazardous environment such as those caused by micrometeorites. The Remote Agent software utilizes model-based reasoning algorithms, constraint-based, goal-directed planning and execution algorithms as well as a fail-operational fault-protection approach. The software includes a planner and scheduler that generate time-based and event-based activities labeled as tokens. The executive in the software makes decisions by taking into account knowledge of the spacecraft's state of health, constraints on spacecraft operations, and the plan from the planner and scheduler. The executive expands the tokens into a sequence of commands that are issued directly to the appropriate subsystems and monitors the response to these commands, and reissues or modifies them if the response is not what is desired.

The need for vehicle level reasoning in aircraft will grow in NextGen and beyond because of the increasing complexity in aircraft and the higher reliance on automation. What follows is an overview of a VLRS for aircraft applications.

3 Scope of Vehicle Level Reasoning Technologies

The primary function of a VLRS is to detect faults and failures at the aircraft level, enable isolation of these faults, and estimate remaining useful life. Consider characteristics of some typical faults arising in some subsystems within an aircraft:

1. [propulsion] Turbine blade erosion is a natural part of turbine aging and wearing of the protective coating due to microscopic carbon particles exiting the combustion chamber. As the erosion progresses over time, this fault manifests itself as increase in fuel flow and *gradual degradation* of engine performance.
2. [avionics/software] Loose wire harness connectors. As connector pins corrode, they make intermediate contact. The corresponding software module that receives this signal registers a series of *intermittent open circuit* faults which may eventually corrupts the navigation software and causes a memory overflow instantaneously.
3. [airframe] Actuator stiction. A sticking actuator changes the dynamic response of a control loop. The feedback action provides some degree of resilience making this problem difficult to detect Eventually, the stiction progresses to a point where the actuator saturates and the control authority is completely lost.
4. [software] This scenario describes a fast progression fault in which the incoming navigation data corrupts the guidance software (see ATSB Investigation report 200503722), which then leads to an incorrect solution. The auto-pilot intervenes and over compensates using the engine thrust. This causes high temperature and high speed events in the engine, leading to cascading problems in the generators and secondary power distribution system. Several auxiliary electronics modules react to the power glitch.

Broadly speaking the VLRS needs to address many different fault scenarios, including:

1. Faults whose severity increases with time. These can be further categorized based on the time constant of this evolution such as incipient, slow progression or fast progression.
2. Binary repeating faults whose repetition increases with time. These can be further categorized based on the time interval between repeats such as constant or increasing.
3. Faults whose effects spread throughout the aircraft with time. These can be further categorized based on the size of this influence such as localized (self contained) or widespread.

As shown in Figure 1, VLRS has to reason across three dimensions – shown as three mutually orthogonal axes labeled time evolution (with extremes labeled fast and slow), impact propagation (with extremes labeled localized and widespread), and symptom persistence (with axes labeled intermittent and constant).

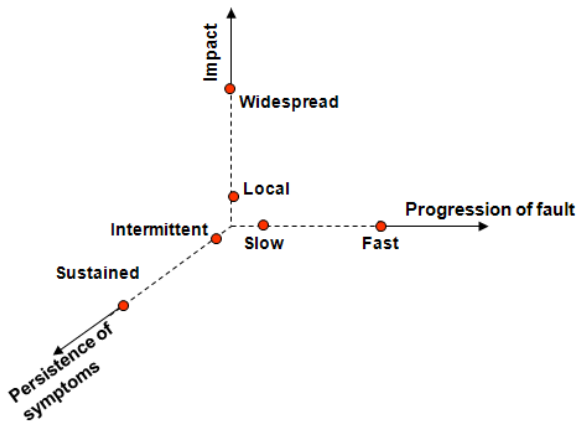

Figure 1. Adverse Events Cube describing the VLRS

reasoning dimensions

Next we describe typical modules that constitute the VLRS. Central to the VLRS are two notions—evidence and failure modes. Broadly speaking evidence represents a symptom or an observation, the failure mode is often an abstract entity that can be mapped to specific corrective or mitigation action. In this respect, a failure mode could map one-to-one with a physical failure like those defined by the component manufacturer, or a condition that has a well-defined corrective action (such as remove and replace) defined in the aircraft maintenance procedures. Symptom evidence takes several forms as we shall discuss in a later section. The four functional modules of VLRS are: inference engine, system reference model, learning loop and communication interfaces. To facilitate the description, the modules are numbered 1 through 4 in **Error! Reference source not found.**.

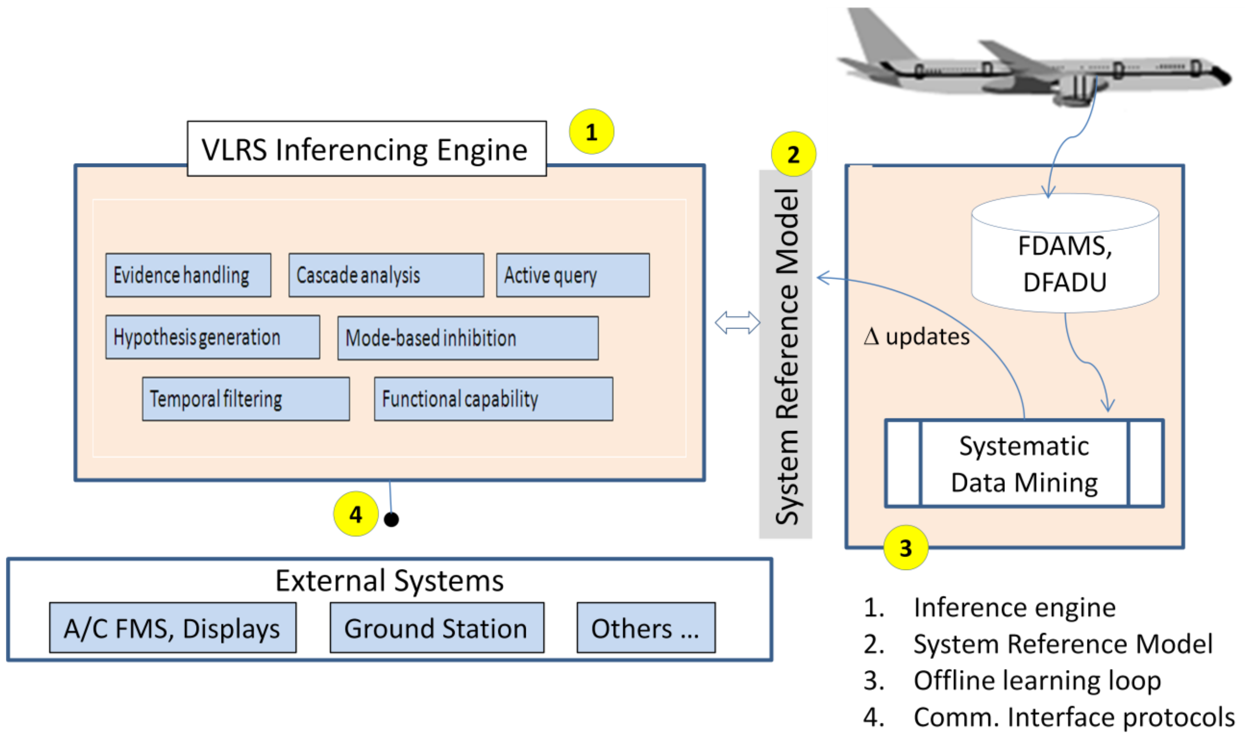

Figure 2. Functional modules in a Vehicle-Level Reasoning System.

1. Interference Engine: This module takes into account health evidence generated from all components and subsystem systems within the vehicle (such as aircraft) to produce the current diagnostic state or predicts the future evolution of a fault. In this process, it produces a most plausible explanation for all the symptoms provided by various components; creates new hypothesis to track multiple faults; deletes hypothesis that may have weak or no evidence support.
2. System Reference Model: The necessary relationships for the inference process are typically separated as a static system reference model. This partitioning allows the same inference engine software code to be reused on multiple vehicles and minimize certification and

qualification costs for deploying VLRS onboard an aircraft. The system reference model, an aircraft loadable software module describes the relationship between evidence generated at the component and/or subsystem level and failure modes that can be mapped to specific maintenance or correction action.

3. Data mining and learning loop: Fleet modeling, data mining and knowledge discovery methods working on historical data can detect anomalies and precursors to critical failure modes. Discovering new patterns and updating old relationships in the system reference model can improve aircraft safety to a higher level continually. Information from this learning loop, resulting in a Δ-change in the reference model enables VLRS to provide accurate health assessment of component, subsystem, or system and support condition-based equipment maintenance and replacement.
4. Communication interfaces: By design, VLRS takes a system-wide view of the adverse event detection problem. While the input interfaces defines how VLRS receives health information from various member components, the output interface defines how it communicates its outputs to the flight crew (displays), ground maintainer (ground station) or a flight management system for automatic fault accommodation.

The overall objective of VLRS is to detect adverse events that may be occurring in the vehicle currently and in some near future. Several functions within this module support this objective. Next we take a closer look at these functions. Figure 3 summarizes them and we describe them below:

To detect events whose characteristics are described in the adverse event cube, the VLRS needs to operate on a variety of evidence generated from several member systems within the vehicle. While the inference engine should not depend on how the evidence was generated, VLRS should support a variety of formats to enable the member system to express their evidence. A monitor is an expression of an evidence. Three common forms of monitors are: (1) diagnostic monitor which is a binary or a probabilistic indication of the evidence being present or absent, (2) prognostic monitor expresses the presence of symptom in future time, and (3) parametric monitor that provides a time series signal together with a threshold whose crossing denotes the presence of a symptom. This is the **Evidence handling** function within the VLRS.

Irrespective of the form in which it is expressed, evidence is always associated with a set of failure modes called its ambiguity group. If there is only one failure mode in this set, which is mapped to a corrective action, and the symptom associated with it is reported, then the current aircraft fault state is isolated. Unfortunately, this is an exception than a rule. Analyzing the overlap between the ambiguity groups of each of observed symptom to postulate an adverse event that can explain most of these symptoms is the hypothesis generation function. Generating and updating the probability associated with these hypothesis based on the strength of observed symptoms is the **Hypothesis handling** function within the VLRS.

VLRS typically reasons with evidence provided by various components or member systems in an aircraft. However, to support future occurrence of adverse events, VLRS needs to take an active role in generation of evidence in addition to a passive, data gathering and analysis activity. In this context, an 'active role' means that the VLRS could generate and test internal hypotheses about the root-cause of a particular adverse event by selecting subsystems and issuing queries designed to verify a hypothesis. In situations where there is contradictory information, a VLRS can disambiguate conflicting health status through passive and active interrogation. This is **active query** function within the VLRS.

Interactions and inter-connection between various subsystems cause a failure mode in one subsystem A to trigger monitors in subsystem B. In addition, a failure mode in subsystem A may cause a failure in subsystem B. These secondary effects provide cascading evidence to the primary common cause failure mode. In other words, VLRS may exonerate subsystem B even though it may be exhibiting an indicting symptom. This is the **cascade** function within the VLRS.

Handling intermittent symptoms is an important function of VLRS. Chatter or intermittency is handled by a simple time-based latching mechanism. Time-based latching requires history keeping and looking for specific temporal patterns such as sinusoids or saw-tooth. Once the pattern is established after analyzing the symptom over a trend window, the VLRS may filter it either to decrease the probability of the prevailing fault hypothesis. This is the **temporal filter** function within VLRS. Often evidence provided by a group of monitors may need to be suppressed when the aircraft is in a specific operating mode. For example, evidence provided by various landing subsystem monitors needs to be suppressed while the aircraft is in a ground engine test mode. We call this the **inhibit** function. While some of this inhibition relationship can be derived based on the power-up state of member systems, the user may also encode additional inhibit conditions for specific aircraft operating modes. While the overall objective of VLRS is to detect and diagnose ongoing failure modes accurately, the net impact of this prevailing fault on mission completion or safety is called **function capability** function.

Often the system reference model builder can derive much of these cascade relationships by following the aircraft wiring and component layout drawings, the user may also encode implicit cascade relationships in the system reference model using past experience.

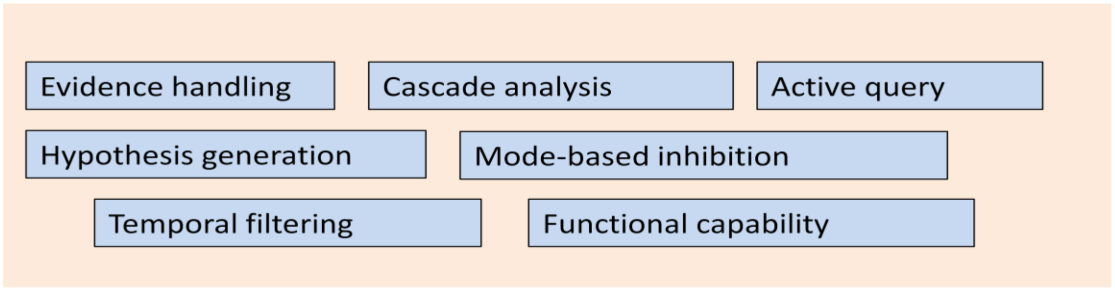

Figure 3. Sub-functions with VLRS inferencing module

4 An Example

The Aircraft Diagnostic and Maintenance System (ADMS) that has been flying on the B777, B787 and business jets is an example of a model-based approach for VLRS. While the ADMS does not embody all

the VLRS functions described earlier, we present this example to highlight some of the technology choices.
The ADMS inferencing engine handles only binary forms of evidence, i.e. a diagnostic monitor. In this format, an evidence has three states: indict (1) or exonerate (0) and a default unknown (-1) state. In the indict state, the evidence asserts that one of the failure modes in its ambiguity set may be occurring in the system. This information $P(fm_j=1|e_i=1)$ is the probability that the evidence e_i will generate an indicting diagnostic monitor when the failure mode ei is present in the system. A bipartite graph is sufficient to capture this relationship, however the system reference model is a network of node entities such as: failure modes, corrective actions, operating modes, evidence, component ID, component supplier and other things needed to generate a maintenance work order automatically.

Relationships to support the cascade analysis function are derived offline by tracking the aircraft component connectivity and layout drawings. In other words, the ADMS implements a topology based cascade analysis; functional and causal cascade are explicitly programmed as additional links in the system reference model network. Temporal filtering is implemented as simple counter-based latching algorithm rather than advanced shape recognition algorithms. ADMS uses a passive interrogation approach to VLRS, that is, no additional data from a particular component or subsystem are collected to improve fault detection and diagnosis.

Since ADMS can handle diagnostic monitors, the overall objective of VLRS, in this case, is to detect the ongoing (at the present time) adverse events that explain all observed symptoms. As described earlier, the ADMS achieves this by generating and updating adverse event hypothesis. Within ADMS, an adverse event hypothesis is tracked using a data structure called fault condition FC. In other words, an FC describes the output generated by the VLRS—namely ADMS in this case. Figure 4 shows the schematic of a fault condition within ADMS.

It has three elements: (1) an initiating event. As we mentioned earlier, ADMS is a passive form of VLRS and hence the inferencing engine is triggered when new evidence in the form of a diagnostic monitor is presented, (2) an ambiguity group is the set of all failure modes that could have triggered this evidence. This is calculated by navigating the system reference model network, (3) an evidence of interest constructed as follow: for each failure mode in the ambiguity set, identify all evidence that could trigger from the reference model. The evidence of interest is constructed by taking a union of all such evidence.

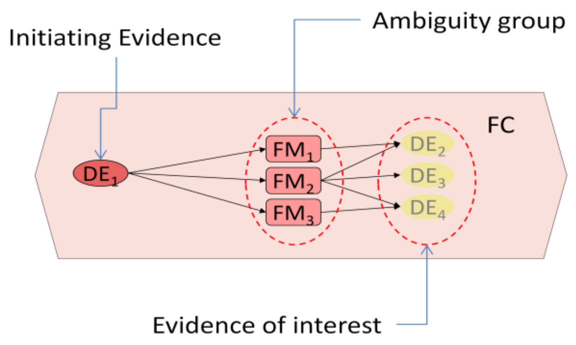

Figure 4. Schematic of a fault condition, which represents the ongoing event hypothesis within the ADMS

The overall probability for the fault condition is defined as the joint probabilities of any of the failure modes in the ambiguity group occurring given the diagnostic monitors (evidence) present. The update is a simple Bayesian update, where the apriori failure mode probabilities are defined in the system reference model.

The inference engine within the ADMS terminates when the failure mode in the ambiguity group reduces to one or a set of failure modes that map to a single corrective action as defined in the system reference model.

5 Case Study

We now provide an example to illustrate the potential safety impact of VLRS. The example surrounds an in-flight engine shutdown (IFSD) incident recorded by the ASIAS database. The flight crew immediately responded and per the operating procedures, turned back and safely landed the aircraft. An in-flight engine shutdown is a highly undesirable adverse event and significantly impacts aviation safety. There were no known exceedances reported for that engine in the recent past, investigation by the maintenance crew indicated a faulty fuel metering unit, which if detected earlier is a routine line maintenance activity—thereby avoiding the IFSD and potential safety impact.

The aircraft was equipped with an aircraft condition monitoring system (ACMS) that records aircraft operational parameters. Evidence provided by several member systems (including the engine) was analyzed by the ADMS. In the previous section we described the ADMS as a first step towards the overall VLRS goals defined in section 3. In this case study, we focus on three VLRS functions (from among many described in Figure 2 and Figure 3) which does not exist in the ADMS. These are: (1) an offline data mining step to discover the causal signature surrounding this adverse event. This analysis could be accomplished based on the parametric data recorded in the ACMS recording system, (2) updating the existing system reference with these newly discovered evidence, (3) expanding ADMS inference engine to generate probabilistic diagnostic monitors for the newly added evidence through the active query function, and (4) adding causal cascade links in addition to the topology-based cascade analysis to the inference engine. While we did not add all the seven functions a VLRS should encode, our objective is to illustrate the improved safety impact by adding the above select features of VLRS. That is, determine if we could detect the underlying fault (failure of the fuel metering unit) several flights before it manifested as the IFSD adverse event. This early notification and the appropriate maintenance action could avoid the safety impact.

We paraphrase the incident report to describe what happened: "After a normal takeoff one of the engines recorded a high-temperature exceedance, which means that the temperature exceeded some pre-specified threshold. The exceedance last long enough to trigger the safety shutdown of the engine."

Message trace from the existing ADMS indicated a fuel metering failure hypothesis was formulated by the inference emgine, however, the probability assigned were too low and competed with other failure hypothesis indicating an engine turbine nozzle failure. In any case, neither one of the hypothesis were strong enough to generate any notification for the flight or the maintenance crew. The evolution of this adverse event was not captured in the system reference model—clearly justifying the need for an offline data mining learning loop to continuously update the system reference to enable the VLRS to detect this fast event.

As we started to analyze this event, it was not clear how a faulty fuel metering unit would have escalated to an over-temperature condition. We employed a standard Tree Augmented Network Bayesian classifier to discover these relationships and hence understand how this specific fault escalated as a safety event. We used the parametric data collected by the ACMS from the last 50 flights leading up to engine shutdown event. Details on the data mining setup are described in [4].

The results are summarized in Figure 5. On the left hand side marked #1 in Figure 5, we summarize the causal sequence of events discovered by the data mining step—starting from the fuel metering actuator fault that initially manifested as sluggish engine start. The controller compensated by aggressive schedules. At some point the controller saturated which resulted in lower idling speeds. Eventually the speed dropped below its allowed threshold triggering while the engine exhaust gas temperature (EGT) remained high triggering the adverse IFSED event.

Second, we encoded this newly discovered knowledge as a Δ-update the system reference model. These updates are marked #2 in Figure 5. This implied adding the following three new evidence to the system reference model: (1) time when the engine started, called the lightoff evidence that looks for abnormally long engine start times, (2) peak exhaust gas temperature evidence that looks for abnormally high exhaust gas temperature during engine startup, (3) the idling engine speed evidence that looks for abnormally high and abnormally low speeds when the engine is idling before aircraft takeoff. The two existing evidences—namely over-temperature exceedance and INFSED are show in Figure 5.

Third, we built the VLRS such that the inference engine exercises all the seven functions we described in section 3. The actual functions exercised by the VLRS are marked #3 in Figure 5. Finally we re-ran the VLRS with the updated reference model using the last 50 flight data before the IFSED event occurred and monitored the outputs—namely the fault conditions as described in section 4 and marked #4 in Figure 5. The most plausible fault state of the aircraft was isolated to a fault condition that contained exactly one element in its ambiguity group—namely the fuel metering unit. Repeated experiments with varying notification threshold on the fault condition hypothesis we concluded that the VLRS would have established the fuel metering root cause anywhere from 20—30 flights before the IFSED event. This would (in theory) allow the maintainer to fix the faulty component, eliminate the source of the fault and hence completely avoid this safety event.

While this single event may be statistically insufficient from a machine learning validation metrics, the clear explanation discovered by the data mining method and the Δ-change to the reference model together with a VLRS seemed sufficient for the engine domain expert to give his thumbs up.

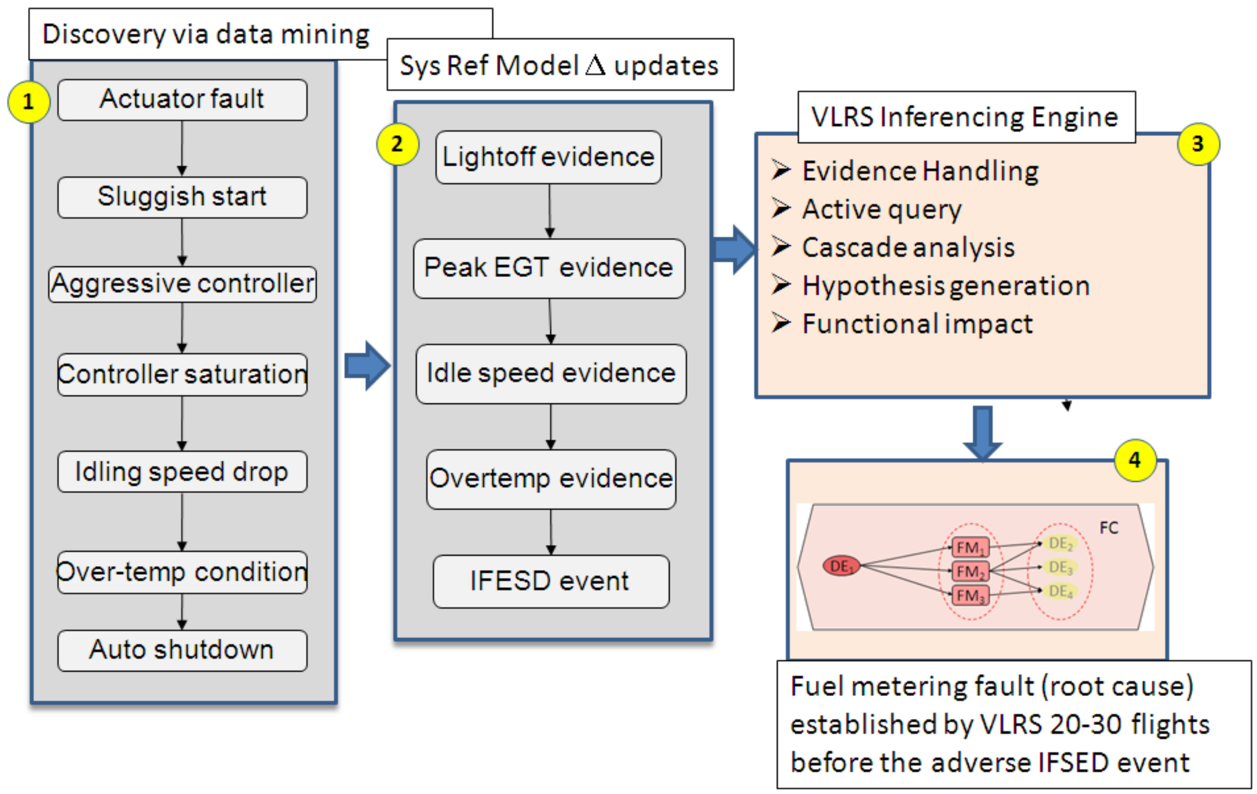

Figure 5. Impact of VLRS on detecting events that led to an in-flight engine shutdown

6 Verification and Validation Issues

From a certification and design assurance perspective, certain incremental VLRS advances are simple extensions to existing health management capabilities that are fielded today. Two examples are monitoring for multi-engine performance asymmetry, and subsystem or component performance trending across multiple flights. In these cases recognized industry standards such as ARP 4761 "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," ARP 4754 "Certification Considerations for Highly-Integrated or Complex Aircraft Systems," and DO-178B "Software Considerations in Airborne Systems and Equipment Certification" may provide sufficient guidance for certification activities pertinent to the VLRS extensions.

A number of V&V issues arise when considering VLRS concepts (such as enriched evidence, tiered computation architecture, and continual data mining-based reference model learning) that are more advanced than component-focused health management systems. Following are just a few of the issues that may arise when considering an advanced VLRS implementation:

- **Large-scale integration and Commercial-Off-the-Shelf (COTS)** - The large-scale integration of health and performance data from subsystems and components that were designed for standalone purposes will require agreement on new VLRS interface standards. Integrating data from COTS-based systems (such as ground-based maintenance history databases or archived vehicle usage data) with airborne systems will raise issues such as guaranteeing the accuracy and integrity of the off-board databases, and addressing the different regulatory requirements that govern the certification of ground-based systems from that of airborne systems.

- **Protecting intellectual property** - Access to increased levels of health and performance information will be required. In multi-vendor environments the resulting increased data exposure may require new paradigms for protecting intellectual property, and current approaches to software configuration, testing suites, and change management (field upgrades) may need to be reexamined.

- **Flight-crew interaction** - In some advanced VLRS concepts, the flight crew collaborates with the health monitoring automation, allowing the VLRS designer to incorporate the knowledge, agility, and diagnostic capability of the flight crew into the analysis algorithms. However, interaction with the flight crew greatly increases the required V&V activities because new hazards, such as increased pilot workload and misleading displays, are introduced into the safety assessment.

- **Interaction with flight controls** - The generation of pre-programmed control surface commands for real-time aerodynamic estimations would provide the VLRS with additional capability to isolate faults and potentially improve overall health state assessment. For example, manually triggered pre-programmed surface excitations may be invoked to identify performance degradation due to ice contamination [5]. VLRS systems incorporating such interrogation capability would have to be subjected to the highest level of V&V scrutiny, because anomalous behavior could lead to a catastrophic failure condition.

- **Non-deterministic techniques** - In-situ data-driven approaches, such as machine learning and certain signal analysis approaches, offer powerful techniques for diagnostic monitoring, prognosis, and characterization of interactions between components and subsystems [6]. The non-deterministic aspects of these approaches will require new V&V tools and methods in order to gain regulatory acceptance, because the civil aviation community does not currently permit the use of such algorithms in high criticality applications.

A fully realized VLRS will be a complex, highly integrated system that employs connectivity and failure models, data-driven and probabilistic methods, and system and subsystem interrogation. The system will require large amounts of data from numerous sources such as onboard flight data, component, subsystem, and system health and performance data, and archived historical data. In order to validate a VLRS capability, real-world data sets must be available that contain well-understood and documented anomalous events, including data archived from flights preceding the anomalous event. Obtaining such data is, however, often impossible due to privacy, proprietary, and legal concerns. The problem will be particularly severe if the VLRS is operating in a multi-vendor environment that will require data sharing among different organizations. Innovative approaches to this problem are needed to support validation activities while also protecting legitimate data sensitivity.

7 Conclusions

The concept of Vehicle-Level Reasoning Systems (VLRS) has its origin in sophisticated diagnostic systems applicable to deep-space applications. In those applications, the VLRS enables a higher degree of autonomy and mission assurance. In aeronautical applications, most modern aircraft are equipped with some degree of vehicle level reasoning. We have discussed the basic architecture of the reasoner and have also shown the different types of data monitors that are applicable to the overall system. These data monitors can generate ambiguity groups that require passive or sometimes active interrogation techniques to help disambiguate the root cause of the adverse event. We have discussed a case study in which a conceptual VLRS could identify the root-cause of a problem nearly 20 to 30 flights ahead of the full manifestation of the problem. The verification and validation of these systems is critical for the future implementations.

8 References

1. Beaming Up a Software Doctor, http://www.astrobio.net/pressrelease/1240/beaming-up-a-software-doctor
2. Deep Space 1: Advanced Technologies: Remote Agent FAQ, http://nmp.nasa.gov/ds1/tech/autoraFAQ.html
3. Tim Felke, George D. Hadden, Dave Miller, Dinkar A. Mylaraswamy "Architectures for Integrated Vehicle Health Management", Proceedings of the AIAA Infotech@Aerospace 2010 Conference, April 2010, Atlanta, Georgia.
4. Dan Mack, Gautam Biswas, Xenofon Koustadas, Dinkar A. Mylaraswamy "Deriving Bayesian Classifiers from Flight Data to Enhance Aircraft Diagnossis Models", Submitted to PHM Society conference 2011.
5. Gingras, D. R., Barnhart, B., Ranaudo, R., Martos, B., Ratvasky, T. P., and Morelli, E., "Development and Implementation of a Model-Driven Envelope Protection System for In-Flight Ice Contamination", AIAA Guidance, Navigation, and Control Conference, 2 - 5 August 2010, Toronto, Ontario Canada
6. Patterson-Hine, A., Narasimhan, S., Aaseng, G., Biswas, G., Pattipati, K., "A Review of Diagnostic Techniques for ISHM Applications." 1st Integrated Systems Health Engineering and Management Forum. Napa, CA. November 2005.