August 2, 2011

Software Health Management with Bayesian Networks

Ole J Mengshoel, *Carnegie Mellon University*
Johann M Schumann

Software Health Management with Bayesian Networks (Extended Abstract)

Ole J. Mengshoel
Carnegie Mellon University
Moffett Field, CA 94035
Ole.Mengshoel@sv.cmu.edu

Johann Schumann
SGT, Inc. / NASA Ames
Moffett Field, CA 94035
Johann.M.Schumann@nasa.gov

***Index Terms*—software health management, fault detection and diagnosis, Bayesian Networks, arithmetic circuits**

I. Introduction

Most modern aircraft as well as other complex machinery are equipped with diagnostics systems for their major subsystems. During operation, sensors provide important status information about a subsystem (e.g., the engine), and this information is used to detect and diagnose faults. Typically, FDDR (fault detection, diagnosis, and recovery) or IVHM (Integrated Vehicle Health Management) systems are used for this purpose. Most of these systems focus on the monitoring of a mechanical, hydraulic, or electromechanical component of the vehicle or machinery. Only recently, health management systems that monitor *software* have been developed (for an overview see, e.g., [1]). In this paper, we will briefly discuss our approach of using Bayesian networks for software health management (SWHM) [2–4].

The field of system health management for hardware is quite mature; this includes the use of Bayesian networks for fault detection and diagnosis [5–12]. Many industrial systems use FDDR systems (e.g., the automotive and aerospace industries). However, the health management of *software* has to adhere to substantially different requirements. One striking difference is that software faults usually occur instantaneously, whereas faults in hardware systems tend to develop over time (e.g., an oil leak).[1] Furthermore, many software problems are caused by software-hardware interactions, which means that both the software and the hardware must be monitored.

At the same time, software has features that might make system health monitoring easier and more promising in some ways. First, the introduction of software redundancy does not increase the weight of a system, while hardware redundancy clearly does. Second, software can be debugged and fixed remotely, without need for human presence at the location where the system is deployed (say, a robotic vehicle on Mars).

Based on the brief discussion above, it is clear that software has several unique features, making a dedicated research and development effort worthwhile. At the same time, it is important to utilize and extend existing results from more traditional FDDR areas. In our SWHM approach, briefly presented here and discussed in more detail elsewhere [2–4], we are using Bayesian networks [13], [14] to define the health model for the software to be monitored. In this extended abstract, we will first discuss SWHM requirements, which make advanced reasoning capabilities for the detection and diagnosis important. Then we will present, at a high level, how our Bayesian SWHM models are constructed and perform.

II. SWHM Requirements

Traditional FDDR (and IVHM) systems are tied to the individual components or subsystems they monitor. Based upon sensor readings, such a system tries to detect, for a component or subsystem, anomalous behavior and if such behavior is found, produces a diagnostic message. While in many cases such an approach is reliable, adverse effects that have been caused by the interaction between different subsystems or components cannot be captured properly. A typical example is a recent incident on a Qantas A380. When one of the engines exploded during flight, taking out the hydraulic system and damaging the wing, the pilots had to sort though literally hundreds of diagnostic messages in order to find out what happened. In addition, several diagnostic messages contradicted each other.[2] If the diagnostics had been system-wide, the number of warnings (and the pilot's workload) could have been reduced tremendously and no contradictory diagnostic messages would have been produced.

The problem of interaction between components or subsystems, as discussed above, is a subclass of a broader class of problems: a specific set of observations could have been caused by a number of different, potentially contradictory faults. The SWHM should be able to distinguish those and provide a metric for how confident the SWHM is that a certain fault has actually occurred.

Many approaches to diagnostics and IVHM use discrete models and do not properly account for sensor failure; diagnostic messages are often produced using table-driven or fault-tree based mechanisms. The input of such systems are most often discretized sensor values (e.g., pressure_low, pressure_hi) and the reasoning uses one or more "firing" diagnostic

[1] This is a rule of thumb, with exceptions. A memory leak, for example, is a type of software fault that develops over time.

[2] See http://www.aerosocietychannel.com/aerospace-insight/2010/12/exclusive-qantas-qf32-flight-from-the-cockpit/

rules. However, those approaches usually do not take into account that sensors, which produce the input to the IVHM, might return noisy data or can be broken altogether. Advanced SWHM, however, should be able to reason about sensor reliability and quality of sensor data.

Finally, for real-time and embedded systems there are requirements for SWHM, like other types of system health management, to have predictable and short execution times and not use much memory [10]. A more general requirement is ease of modelling, either by supporting machine learning or automated Bayesian network construction from a domain-specific language.

III. Bayesian Networks for SWHM

Bayesian networks are an approach to represent multivariate probability distributions in a compact manner such that they are amendable to learning and inference [13], [14]. We have successfully compiled FDDR Bayesian networks to arithmetic circuits [9–11], which are then used to perform, using an on-line evaluator, system health management functions including detection and diagnosis.

In our approach, we use Bayesian networks to model the nominal and off-nominal behavior of software [2]. Our modelling of software is inspired by previous work on system health management for electrical power systems [9], [10], [11] in which each electrical power system component is represented by a small number of nodes (typically 2–6), and then separate Bayesian networks structures represent the connections between components. In a similar way, each software component is in our approach represented by a small number of nodes, one of which represents the "health status" of the software.

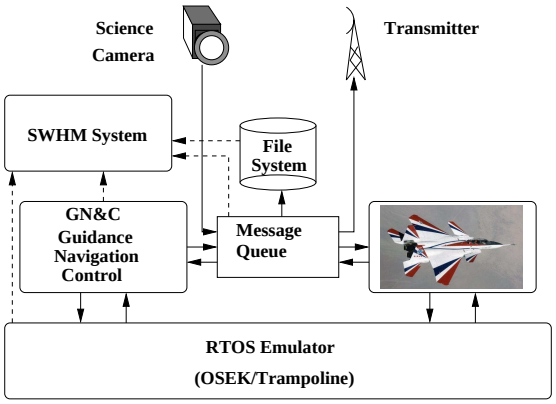

Fig. 1. Demonstration Software Architecture

Let us consider the following, highly simplified example: The GN&C software of an aircraft (Figure 1) communicates with the aircraft's sensors and actuators using a global message queue (see [4] for details). This queue is also used to route data from a science camera to a transmitter, which downlinks the images. Furthermore, all messages going through the message queue are being logged to an on-board file system using blocking writes.

In a nominal mode, the system works fine. Now suppose that the file system on-board the aircraft is full or almost full. The given software design causes control messages to stay longer in the message queue. They may even be dropped, because of the time it takes to write into a full or almost full file system. Furthermore, delayed control messages can cause oscillation of the entire aircraft, a potentially dangerous situation, which can even lead to loss of the vehicle. In a similar manner, a larger than expected number of images from the science camera could cause the message queue to overflow even under "normal" circumstances; and the controller and science camera could (if not designed properly) compete for slots in the queue, thereby causing delayed or dropped messages.

The goal of our SWHM system is to correctly diagnose such situations. We have implemented a simple SWHM demonstration system using OSEK[3] as the underlying operating system. The plant model, the GN&C software (including message queue and file system), as well as the arithmetic circuit SWHM model evaluator, are executed as individual processes. While simplified compared to a real-world system, our demonstration system captures many of its key attributes.

In an experimental study, the SWHM system indicated no faults in nominal runs. On the other hand, when the simulation started with an almost full file system as discussed above, the aircraft started oscillating and the SWHM system correctly diagnosed the root cause (Figure 3). Our model relies on the use of an oscillation BN evidence node, which uses a Fast Fourier Transform to detect vibrations and oscillation. At the same time, the health probabilities for the pitch and accelerometer systems are generally high (albeit with some transient lows). Overall, the SWHM correctly points to a software fault as the most likely root cause of the oscillations, given the available evidence (at around $t = 110$).

Further details on this approach to software and sensor monitoring, applied to small satellites and using scenarios with injected faults, are available in [2].

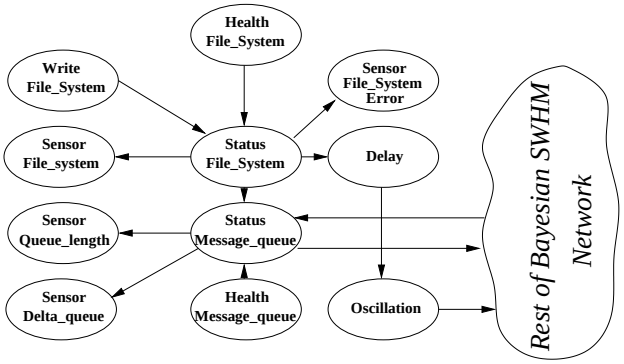

Fig. 2. Partial Bayesian network for file system related architecture.

In our demonstration, we have implemented the SWHM concept using Bayesian networks, which model software as well as interfacing hardware sensors. After compilation to

[3]Trampoline/OSEK, http://trampoline.rts-software.org

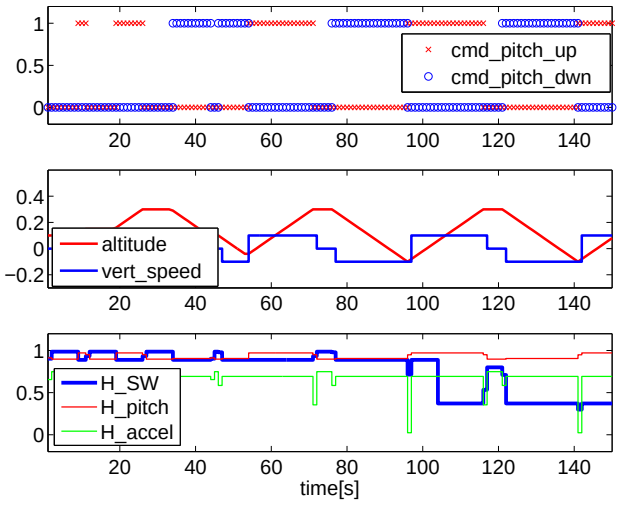

Fig. 3. Temporal trace for file system related fault causing oscillations

arithmetic circuits, Bayesian networks are well-suited for on-line execution in embedded software systems found in vehicles (aircraft, spacecraft, and cars) or mobile devices (cell phones, tablets, etc.) [9–11]. This approach can fuse information from different layers of the software stack, from firmware and operating system to application software.

IV. Conclusion

Software plays an important and increasing role in aircraft and other complex machinery. Unfortunately, software can fail in spite of extensive verification and validation efforts. In this paper, we discussed a Software Health Management (SWHM) approach to handle software bugs and failures. We have briefly presented an SWHM system that can help to perform fault detection and diagnosis in embedded systems, using Bayesian networks as the underlying modeling paradigm. In these networks, we concisely capture and fuse information from hardware sensors, software status signals, software quality signals, and information from the operating system. Given these data, Bayesian reasoning can compute the most likely causes of failures, if present.

A Bayesian network system health model can be compiled into an efficient arithmetic circuit, which yields a high-performance SWHM that is suitable for execution within embedded (on-board) software systems, and amenable to V&V [15]. Furthermore, mature software tools for Bayesian network modeling and compilation into arithmetic circuits—such as SamIam[4] and Ace[5]—are readily available.

In this abstract, we only covered a small range of an SWHM system's capabilities. Current work investigates, how information on the quality of a computation (e.g., numerical quality or quality of the state estimation) can be smoothly incorporated into the SWHM. Research on hierarchical SWHMs will address the issue of detecting complicated software-hardware interactions for large- and extreme-scale Bayesian networks [16], and will focus on the fusion of multiple information streams for the purpose of increasing diagnostic accuracy. Finally, future work will investigate how our SWHM approach can deal with unexpected and unmodeled failures (e.g., due to unforeseen environmental circumstances) and emerging behavior. Bayesian networks have—due to their modeling capabilities, efficient execution, and high reasoning power— a great opportunity to to find their way into on-board software health management.

[4]http://reasoning.cs.ucla.edu/samiam/

[5]http://reasoning.cs.ucla.edu/ace/

Acknowledgment

This work is supported by a NASA NRA grant NNX08AY50A "ISWHM: Tools and Techniques for Software and System Health Management".

References

[1] G. Karsai, Ed., *1st International Workshop on Software Health Management (SMH 2009)*. ISIS, Vanderbilt University, 2009. [Online]. Available: http://www.isis.vanderbilt.edu/workshops/smc-it-2009-shm

[2] J. Schumann, O. Mengshoel, and T. Mbaya, "Integrated software and sensor health management for small spacecraft," in *Proc SMC-IT*, 2011.

[3] A. Srivastava and J. Schumann, "The case for software health management," in *Proc SMC-IT*, 2011.

[4] J. Schumann, T. Mbaya, and O. Mengshoel, "Bayesian software health management for aircraft guidance, navigation, and control," in *Proc. PHM 2011* , 2011.

[5] C. Skaanning, F. V. Jensen, and U. Kjærulff, "Printer troubleshooting using Bayesian networks," in *Proc IEA/AIE '00: Proceedings of the 13th international conference on Industrial and engineering applications of artificial intelligence and expert systems*. Springer-Verlag, 2000, pp. 367–379.

[6] U. Lerner, R. Parr, D. Koller, and G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems," in *Proceedings of the Seventeenth national Conference on Artificial Intelligence (AAAI-00)*, 2000, pp. 531–537. [Online]. Available: citeseer.ist.psu.edu/lerner00bayesian.html

[7] C. Romessis and K. Mathioudakis, "Bayesian network approach for gas path fault diagnosis," *Journal of engineering for gas turbines and power*, vol. 128, no. 1, pp. 64–72, 2006.

[8] O. J. Mengshoel, A. Darwiche, K. Cascio, M. Chavira, S. Poll, and S. Uckun, "Diagnosing faults in electrical power systems of spacecraft and aircraft," in *Proceedings of the Twentieth Innovative Applications of Artificial Intelligence Conference (IAAI-08)*, 2008, pp. 1699–1705.

[9] B. W. Ricks and O. J. Mengshoel, "Methods for probabilistic fault diagnosis: An electrical power system case study," in *Proc. of Annual Conference of the PHM Society, 2009 (PHM-09)*, 2009.

[10] O. J. Mengshoel, M. Chavira, K. Cascio, S. Poll, A. Darwiche, and S. Uckun, "Probabilistic model-based diagnosis: An electrical power system case study," *Systems, Man and Cybernetics, Part A: Systems and Humans*, IEEE, vol. 40, no. 5, pp. 874–885, 2010.

[11] B. W. Ricks and O. J. Mengshoel, "Diagnosing intermittent and persistent faults using static Bayesian networks," in *Proc. of the 21st International Workshop on Principles of Diagnosis (DX-10)*, 2010.

[12] A. Choi, A. Darwiche, L. Zheng, and O. J. Mengshoel, "Tutorial on Bayesian Networks for System Health Management," in A. Srivastava and J. Han, Eds. *Data mining in systems health management: Detection, diagnostics, and prognostics*, Chapman and Hall/CRC Press, 2011.

[13] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[14] A. Darwiche, *Modeling and Reasoning with Bayesian Networks*. Cambridge, UK: Cambridge University Press, 2009.

[15] J. Schumann, A. Srivastava, and O. Mengshoel, "Who guards the guardians? — toward V&V of health management software," in *Runtime Verification 2010*. Springer, 2010.

[16] K. Przytula, G. Isdale, and T.-S. Lu, "Collaborative development of large Bayesian networks," in *Proc. of the 2006 IEEE Autotestcon*, 2006, pp. 515–522.