Detecting and Diagnosing Faults

Bernhard Rinner
Institut für Technische Informatik,
Technische Universität Graz

The world is full of faults! Some devices develop faults once they are put into use; some devices have faults from the moment of manufacture. Some faults lead to the unusability of a device or even to catastrophes [6]; some faults can be ignored or are actually never discovered. There is, therefore, an increasing demand for mechanisms that automatically detect, isolate and understand faults in a „supervised" system in order to improve the system's reliability and to avoid possible damage and breakdown. Fault detection and diagnosis mechanisms are now included in applications such as process engineering, power engineering, automotive engineering as well as aerospace and robotics and are deployed in large plants and small handheld devices. In this article, I briefly introduce the basic approaches to monitoring and diagnosis focused on techniques used in engineering applications and technical systems. A short discussion on challenges for monitoring and diagnosis concludes this article.

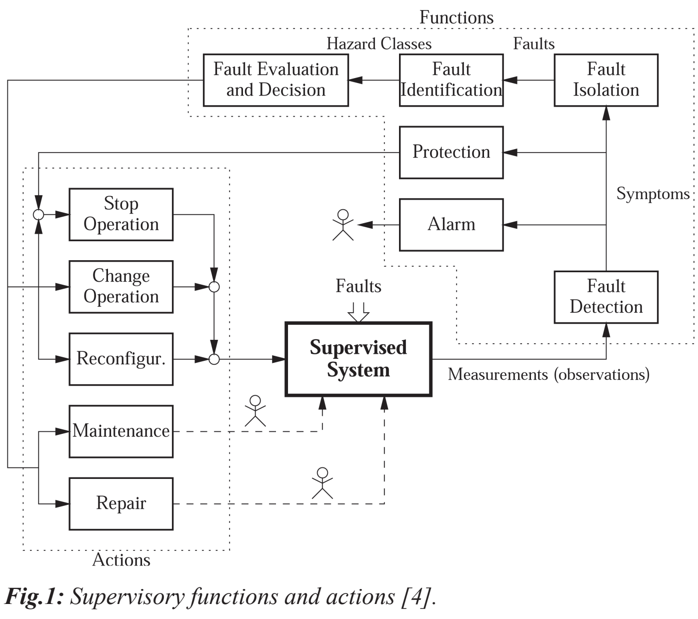

***Fig.1:** Supervisory functions and actions [4].*

Supervision of Technical Systems

Monitoring and diagnosis are instances of supervision. The individual steps of supervision can be grouped into *functions* and *actions*. Functions perform some kind of evaluation given the measurements (observation) and some a priori knowledge about the supervised system. Actions perform some modifications on the supervised system to maintain the operation in case of faults. Figure 1 presents an overview of the different supervisory functions and actions.

Fault detection (monitoring) checks the current state of the supervised system and makes a binary decision as to whether something has gone wrong or everything is working fine. In the case of a fault, it generates an *alarm* to the user/ operator. In the case of a dangerous state of the supervised system, *protection* initiates an appropriate counteraction. Based on the information derived from fault detection, *fault isolation (diagnosis)* determines the location of the fault, i.e., which component has become faulty. In addition to fault isolation, *fault identification* estimates the size and type or nature of the fault. This function is essential for most supervisory actions (see below).

Depending on the supervisory level of the technical system different actions can be performed. A *stop operation* tries to abort a dangerous evolution of the technical system into a serious malfunction. In the case of a fail-safe technical system, the stop operation automatically initiates a transition into a fail-safe state of the technical system. A *change operation* automatically transfers the technical system into a new point of operation, e.g., by changing the set point of its controller. *Reconfiguration* automatically alters the configuration of the technical system. This may include the activation of a different controller and the usage of different components of the technical system. Reconfiguration may further change the goal of the technical system. The user/ operator manually checks individual components of the technical system. *Maintenance* is often initiated at predefined times in order to prolong the normal operation of the technical system. Typically, the technical system remains operational during maintenance. The user/ operator physically replaces faulty components. *Repair* is initiated after a failure has occurred. Normally, the technical system is not in (full) operation during repair.

Approaches to Monitoring

As depicted in Figure 1 all supervisory functions are grounded on the detection of faults. Fault detection decides as to whether the system is working as expected or not. In order to make this decision, we have to know *what is expected* from the technical system. Thus, we need some kind of redundant information which can be exploited to make this decision.

A (simple) method for fault detection is *limit checking*, i.e., to compare the measured signal to the nominal level or trend of this signal. The nominal properties of the signal have to be known a priori, e.g., by gathering measurements from a healthy technical system. While this method is simple to implement, it has severe drawbacks such as the possibility of false alarms in the event of noise, input variations and the change of the operating point. Another drawback is that a single fault could cause many system signals to deteriorate from their nominal level and appear as multiple faults.

Another method to fault detection is based on *hardware (or physical) redundancy*. Multiple lanes of sensors, actuators, computers and software are used to measure and/or control a particular variable. A voting scheme is typically applied to decide whether a fault has occurred and its likely location amongst redundant system components. Hardware redundancy is common in safety-critical systems such as fly-by-wire control systems and nuclear reactors. The major drawback of hardware redundancy is the extra equipment and maintenance cost as well as the additional space and energy required.

In the *analytical (functional) redundancy* method, a mathematical model of the technical system is used to generate the dissimilarly measured variable rather than replicating each hardware individually. In this approach, redundant analytical relationships between various measured variables of the supervised technical system are exploited.

Analytical redundancy makes use of a mathematical model of the supervised technical system and is, therefore, often referred to as the *model-based approach to fault diagnosis*. Fault detection in model-based diagnosis is normally achieved through a comparison between a measured signal with its estimation. The estimation is generated by a mathematical model of the system being considered. Whenever there is discrepancy between observation and prediction a fault has been detected (Figure 2).

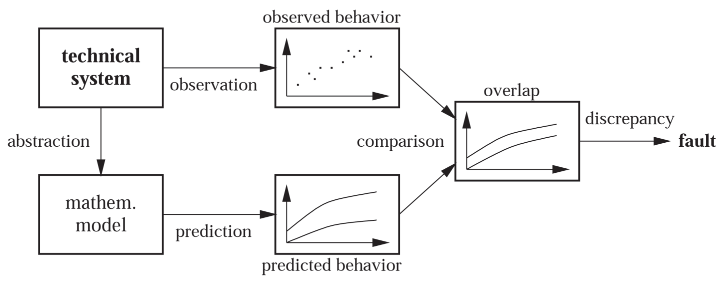

***Fig.2:** Model-based Monitoring compares the observed and predicted behavior of a technical system.*

Approaches to Diagnosis

Approaches to diagnosis of technical systems have been investigated in different research communities over a long period of time. Artificial intelligence (AI) as well as process and control engineering are probably the most influential communities that have performed research in these areas. While researchers in artificial intelligence as well as process and control engineering have developed (independently) *model-based approaches* to diagnosis [3], the focus of interest as well as the techniques applied are quite different. The major objective in process and control engineering is to improve the reliability, availability and safety of technical processes. The applied *fault detection and isolation (FDI)* methods are essentially based on numerical methods such as state estimation and parameter estimation [1]. Inspired by human reasoning, research in AI has aimed to solve diagnosis by exploiting only qualitative properties of the technical system. In general, the technical system is described using logical formulae, and reasoning based on logical rules is used to solve the diagnosis problem.

Finding diagnoses can also be seen as a kind of classification where given the system's input and output the goal is to classify the system's state to predefined fault classes. *Associative diagnosis* is a form of heuristic classification which originates from early expert system research. Associative diagnosis systems are built by accumulating the experience of expert diagnosticians in the form of empirical associations. The expert knowledge is simply encoded by rules that express the relationship between symptoms and diagnoses. This means that no (causal) model of the supervised system is required for generating a diagnosis. Associative diagnosis has been quite successfully applied in expert systems, especially in the medical domain. However, a major drawback of this approach is the knowledge acquisition. Encoding the expert knowledge into rules is a tedious task and must be performed from scratch for each new application. In addition, it is not easy to check the rules for consistency.

In the *logic-based approach* to model-based diagnosis, logical formulae represent both the structure and behavior of the supervised system as well as the set of observations characterizing a specific instance of a diagnostic problem. Logic-based diagnosis aims to formally and unambiguously deduce the solutions to a diagnostic problem expressed in these formulae using well-defined concepts in the underlying logical framework.

Reiter [7] was among the first who developed a theory of diagnosis based on a logical framework. In this very general theory of diagnosis, first-order logic is used to represent task specific information. Further, a domain-independent concept of a system is introduced to formalize as abstractly as possible the concept of a component and the concept of a collection of interacting components. De Kleer and Williams [5] introduced an inference method for generating all (minimal) diagnoses that can be derived from this abstract specification. The implementation of this diagnosis algorithm is called *general diagnosis engine (GDE)* which serves as a root for many diagnosis applications.

Fault detection and isolation (FDI) compares the available measurements from the supervised system with a priori information represented by the system's mathematical model in order to detect and isolate faults in components of the system. Most FDI methods are grounded on generating and evaluating *residuals*. A residual is a fault indicating signal generated from the difference between real measurements and estimates of these measurements using the mathematical model. Various residuals can be designed with each having special sensitivity to different faults. The subsequent analysis of each residual, once its threshold is exceeded, then leads to fault isolation. The general structure of FDI consists, therefore, of the two consecutive steps: (i) *residual generation* and (ii) *decision making* which determines which fault(s) might have occurred given the set of residuals (Figure 3).

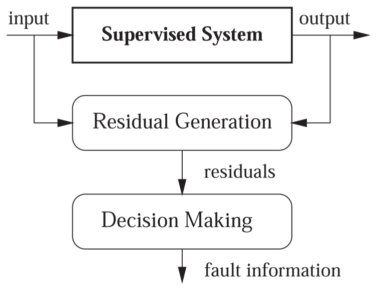

***Fig.3:** General structure of FDI.*

Challenges and Future

Monitoring and diagnosis have been successfully deployed in a variety of engineering applications. However, there are still some challenges for monitoring and diagnosis, especially when applied in technical environments [8].

First, most approaches are based on some models of the supervised system, and the quality of the models is very important for the overall

performance. However, building appropriate models is generally tedious and expensive. Support for *generating, deriving and re-using models* are becoming more and more important.
Second, the supervised system is in most cases not completely known, e.g., due to device tolerances, and exact parameters and functional relations for a model may not be specified. The observation can only provide an incomplete view on the supervised system due to discrete sampling, limited observability and noise. The monitoring and diagnosis system must, therefore, be able to *reason with incomplete and uncertain information*.
Third, the monitoring and diagnosis system is coupled to the supervised system via sensors and actuators and must react within predefined time windows. *Real-time operation* is often required.

Automated monitoring and diagnosis are also becoming more and more important for commercial products, and these techniques will eventually influence everyday life. Monitoring and diagnosis capabilities – whether visible to the user or not – are incorporated into many applications and systems in many different domains. The increased interest on this technology is demonstrated by many activities in industry and academia. Examples for such activities include dedicated conferences [2][9] and networks of excellence [10]. We will, therefore, look into a bright future for monitoring and diagnosis.

References

[1] Jie Chen and Ron J. Patton. Robust Model-Based Fault Diagnosis for Dynamic Systems. Kluwer Academic Publisher. 1999.
[2] A.M. Edelmayer (Editor). Proceedings of the 4th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS '2000), Budapest Hungary, 2000.
[3] Walter Hamscher, Luca Console, and Johan de Kleer (Ed.). Readings in Model-based Diagnosis. Morgan Kaufmann, 1990.
[4] Rolf Isermann. Supervision, Fault-Detection and Fault-Diagnosis methods: An Introduction. Control Engineering Practice, 5(5):639-652, 1997.
[5] Johan de Kleer and Brian C. Williams. Diagnosing Multiple Faults. Artificial Intelligence, 32:97-130, 1987.
[6] Charles Perrow. Normal Accidents. Princeton University Press, ISBN: 0691004129, 1999.
[7] Raymond Reiter. A Theory of Diagnosis from First Principles. Artificial Intelligence, 32:57-95, 1987.
[8] Bernhard Rinner. Monitoring and Diagnosis of Technical Systems. Habilitationsschrift, TU Graz, 2001.
[9] Markus Stumptner and Franz Wotawa (Editors). Proceeding of the International Workshop on Principles of Diagnosis (DX'02). Semmering, Austria, May 2002.
[10] MONET. European Network of Excellence in Model-based Systems & Qualitative Reasoning. http://monet.aber.ac.uk ■

Bernhard Rinner, Institute for Technical Informatics, Graz University of Technology, Inffeldgasse 16, 8010 Graz, rinner@iti.tu-graz.ac.at