# Fault Diagnosis with Adaptive Control for Discrete Event Systems

**Yamen El Touati** [†] **and** **Mohamed Ayari** [††]

[†]Department of Computer Sciences,
[††]Department of Information Technology
Faculty of Computing and Information Technology,
Northern Border University, Kingdom of Saudi Arabia.

**Summary**
Discrete event systems interact with the external environment to decide which action plan is adequate. Some of these interactions are not predictable in the modelling phase and require consequently an adaptation of the system to the metamorphosed behavior of the environment. One of the challenging issues is to guarantee safety behavior when failures tend to derive the system from normal status. In this paper we propose a framework to combine diagnose technique with adaptive control to avoid unsafe sate an maintain the normal behavior as long as possible.

***Key words:***
*Diagnose, Discrete event systems, Control, Finite State Machines*

## 1. Introduction

When modeling systems, it is important to reason about faulty behavior issues, particularly knowing that fault occurrence may lead to serious damage. Indeed, in order to design resilient systems, we have to think not only about how to detect faults [1, 2, 17], but also about how to react to them. The adaptive control can be seen as an alternative to answer the question "what to do when a fault occurs?". Actually, adaptive systems are reconfigurable systems which are able to respond to environmental changes or interacting changes by behavior reconfiguration [19]. However, before thinking about how to react to fault occurrence, we have to answer a more important question, that is: how to detect faults efficiently? In this context, fault diagnosis, consists of detecting faulty system behavior, localizing its origin and identifying its causes [3, 4, 5, 6], can widely help with this issue. Failure detection and identification can be state-based [11, 12, 14], language based [3, 4, 5, 6, 15], or Petri nets based [13].

Control theory [7, 10, 19] answer to the question "how to force the system respect the safety requirement?". Indeed the controller is able to force some controllable events to avoid any unsafe state [20, 21]. The plant system generates some events that correpond mostly to sensor feedback and are uncontrollable by nature [23, 24]. Besides, the controller action have to be optimal [22] in a way that it performs the least restrictions to the system activities.

The main issue is to recover from errors when they are not tractable [14, 17, 18]. In this case, the controller has no information to detect reaching any unsafe state. Thus, it is interesting to combine the controller with diagnoser action in order to detect any system failure and allow the controller to switch the system to desired safety specification.

In this paper, we propose a frmework that benefits from diagnoser detection and identification of failure that are unobservable sothat the controller guide the system via controllable action events to exit any unsafe state and respect as long as posible the safety specification of the system behavior.

This paper is organized as follows. In the next section, we characterize basic concepts and notations to be used. In section 3, tecniques for diagnose are presented and studied. Section 5 presents the control theory using finite state machines. In section 5 we illustrate the proposed technique for adaptive control combined with diagnosis. We conclude in section 6.

## 2. Basic Concepts and Notations

The following notations are adopted. Σ represents a finite set of actions or events. $\Sigma^*$ represents an infinite set of all possible strings (sequence or concatenation of actions) built with symbols from Σ.

Modelling discrete event systems (DES) frequently uses formal languages or finite state machines. In this work, we use finite state machines (FSM) according to the following definition.

Manuscript received November 5, 2021
Manuscript revised November 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.11.22

A finite state machine is the tuple $P = (Q, \Sigma, \Delta, q_0)$ where

- $Q$ is a finite set of states
- $\Sigma$ is a finite set of events (or actions)
- $\Delta \subset Q \times \Sigma^* \times Q$ is a finite set of transitions
- $q_0 \in Q$ is the initial state

An FSM is nondeterministic where more than one action may be possible in the same state. This allow the FSM to model a lager class of DES.
In Figure 1, an example of FSM is presented.

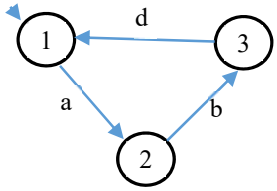


**Fig. 1**: Example of DES modelled by FSA

In this example, the FSM is described by the following information.

- $Q = \{1, 2, 3\}$
- $\Sigma = \{a, b, d\}$
- $\Delta = \{(1, a, 2), (2, b, 3), (3, d, 1)\}$
- $q_0 = 1$

## 3. Techniques of Failure Diagnose

In general, diagnose technique is used to determine when it is possible if the system is in a faulty state. When there is a certainty that the system is in a faulty state, the controller attempt to guide the system to exit failure status and reach a safe state. The diagnoser observes events that are generated by the system to identify the current state. However, there are some issues that make this task more complicated when some of the events are not observable by the diagnoser.

The main goal of a diagnoser is to detect any abnormal behavior and identify the origin of the failure. A failure is considered as misfunction of the system that derives from normal behavior and generates consequently errors. This means that an error is a direct consequence of a failure.

The occurrence of a fault is usually associated with the emission of a set of signals from the system sensors. These signals represent the symptoms that allow the detection and the identification of the error.

The framework architecture of diagnoser is represented in Figure 2. Lower layer corresponds to the physical process or system operational component while the upper layer is the supervision component of the system. Communication between both layers is possible via an interface which transfer command actions form the upper layer to the physical system, and forward visible sensor signals from the physical process to the supervisor. These visible signals are called observable events. The role of the supervisor is to use its computational power with the feedback of the lower layer to detect and identify any possible failure.

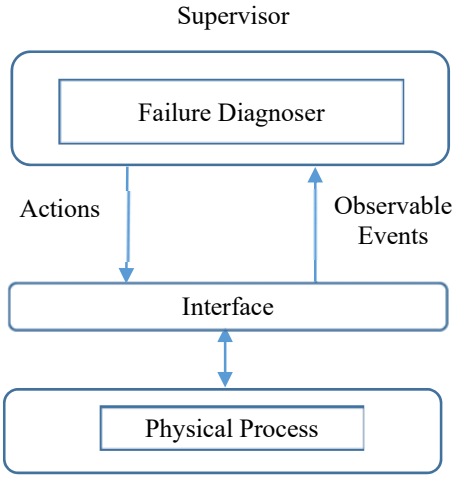


**Fig. 2**. failure diagnosis architecture

Usually, events that are the direct cause of failure are unobservable. This makes the task of the diagnoser more complicated and require an estimation of all possible states of the system in order to deduce the faulty behavior when it happens.

We use the following notations

- $\Sigma$ is the set of all events
- $\Sigma_{uo} \subseteq \Sigma$ is the set of unobservable events
- $\Sigma_o \subseteq \Sigma$ is the set of unobservable events.
- $\Sigma_f \subseteq \Sigma_{uo}$ is the set of failure events

**Remarks:**

- $\Sigma$ is portioned into $\Sigma_{uo}$ and $\Sigma_o$. This means that $\Sigma_{uo} \cup \Sigma_o = \Sigma$ and $\Sigma_{uo} \cap \Sigma_o = \phi$.
- Failure events are unobservable since observable failure events are easy to detect immediately when they occur.

The failure detection and identification are based on the comparison between observable behavior from physical process and expected behavior from system model [1, 2, 3, 6] in order to analyze the discrepancies between these two behaviors and deduce possible errors as shown in Figure 3.

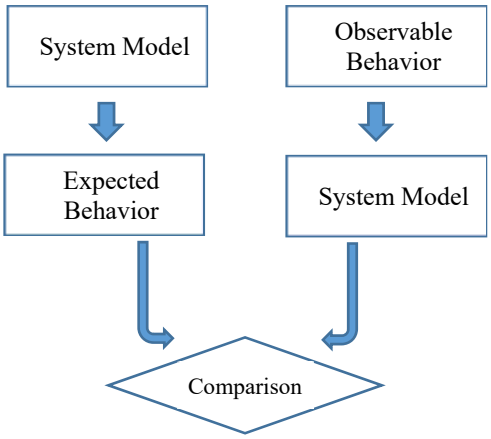


**Fig. 3 :** Model-based Diagnose

The main idea is to design a framework model of the process behavior with finite state machine or any equivalent model. This model describes the normal behavior as well as the failure behavior of the system. Therefore, the diagnoser builds a state estimator according to the received observable events in a form of finite state machine. This process is depicted in Figure 4.

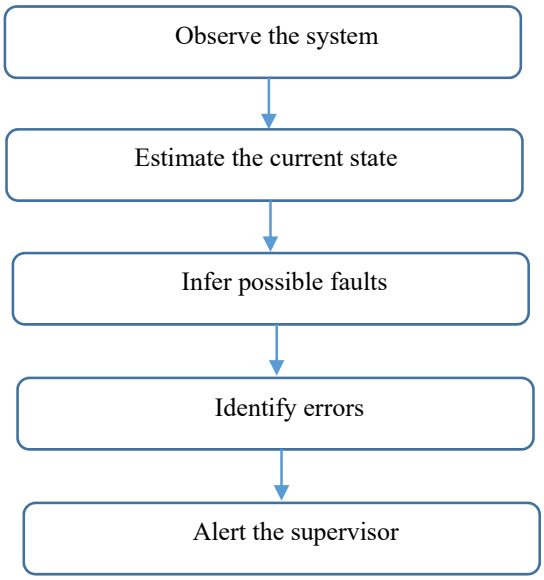


**Fig. 4:** Diagnoser tasks

The Figure 5 illustrates an example of discrete event system modelled by finite state machine. This DES is represented by the following action sets.

- $\Sigma = \{a, b, c, d, f1, f2\}$
- $\Sigma_{uo} = \Sigma_f = \{ f1, f2\}$ is the set of unobservable events
- $\Sigma_o = \{a, b, c, d\}$ is the set of unobservable events.

According to $\Sigma_f$, transitions labeled with *f1* and *f2* lead to failure states. This means that the states 4, 5 and 6 corresponds to failure behavior.

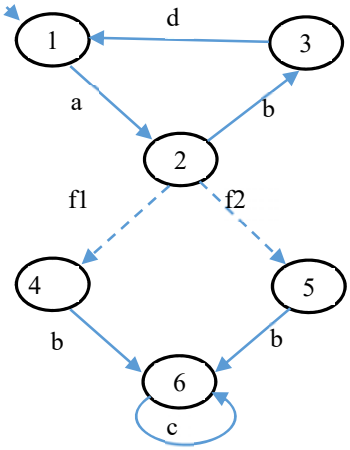


**Fig. 5**: Example of DES modelled by FSA

The diagnoser is represented by the FSM in Figure 5. The initial state of the diagnoser FSM and is labeled by *N* indicating a normal behavior. The sate labeled *2N 4F 5F* indicates that the system may be in state 2 (with normal behavior) or in states 5 or 6 (with failure behavior). If a diagnoser state is labeled exclusively *F*, the state is qualified ***Fi-certain***. By opposition, if the diagnoser state is labeled exclusively *N*, the state is qualified ***Normal***. Otherwise, the state is qualified ***Fi-uncertain***.

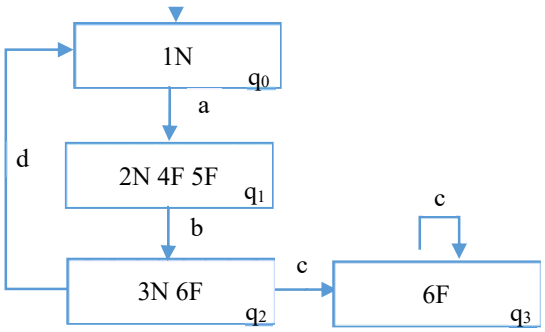


Fig. 6 : Diagnoser FSM

The Table 1 illustrate the diagnoser state qualifications according to failure certainty.

**Table1**: Failue certainty of the diagnoser

| Diagnoser state | Failue certainty |
|---|---|
| $q_0$ | *Normal* |
| $q_1$ | *Fi-uncertain.* |
| $q_2$ | *Fi-uncertain.* |
| $q_3$ | *Fi-certain.* |

Note that no failure is detected if a diagnoser cycle does not contain any Fi-certain state.

## 4. Control of Discrete Event Systems

The supervisory control theory is based on Ramage & Wonham theory [10, 20, 23, 24]. The main idea is to inhibit the production of any action event that may lead to unsafe state as mentioned if Figure 7.

Typically, the controller is able to produce only controllable actions (referred to as $\Sigma_c$). Events that are not managed by the controller are uncontrollable (referred to as $\Sigma_{uc}$) and corresponds usually sensor feedback. The control Algorithm is based on FSM [25]. Each state is considered forbidden if it does not meet the safety specification. Besides, any state that can lead to a ***forbidden*** state by a sequence (one or more) of uncontrollable events is considered to be ***weakly forbidden***. The controller is constructed by truncating any possibility to reach forbidden states and weakly forbidden states by acting on controllable events to avoid (weakly) forbidden states and obtain the most permissive controller.

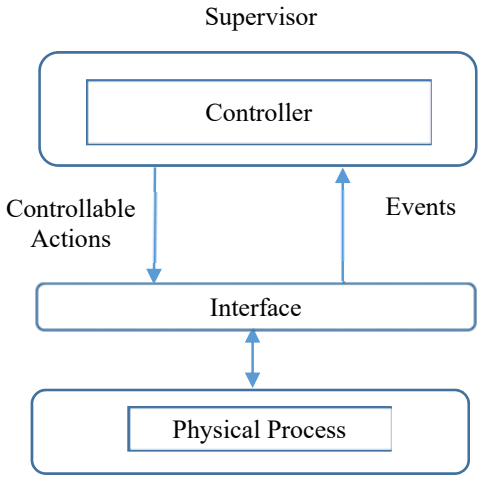


**Fig. 7.** Supervisory control architecture

Note that $\Sigma$ is partitioned into $\Sigma_c$ and $\Sigma_{uc}$. This means that in controller perspective, an event is either controllable or uncontrollable. The complexity of controller action resides in avoiding weakly forbidden states since they may lead spontaneously to forbidden state without any reaction from the controller. A system is considered to be controllable when the controller succeeds to avoid forbidden without reducing the scope of the system. In some cases the controller is brought to reduce some important and critical behavior. In this case, the issue is reported to the system designer which have to take the necessary actions in redesigning to avoid any misbehavior.

## 5. Adaptive Control combined with Failure Diagnosis

It is interesting to combine the computational power of the diagnoser with controller actions to seek safe and normal behavior. The system architecture is represented in Figure 8. The diagnoser reports any failure behavior to the supervisor, then the controller consider all states reported by diagnoser as forbidden states and apply the control algorithm to avoid reaching those states.

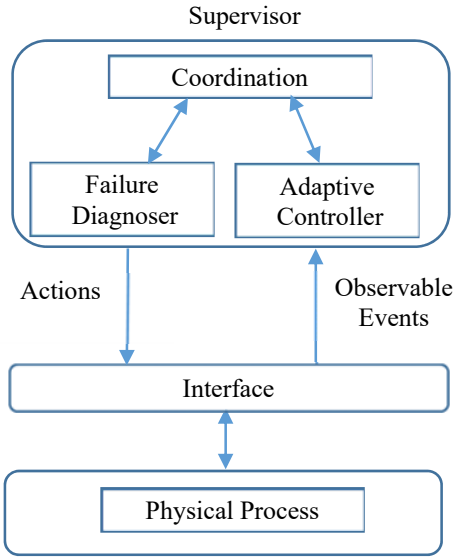


**Fig. 8**. Supervisory control architecture

Consider that

- $P = (Q, \Sigma, \Delta, q_0)$ is the FSM of the physical process model
- $D = (Q^D, \Sigma_o, \Delta^D, q_0^D)$ is the FSM of the diagnoser
- $F \subset Q$ is the set of forbidden states
- $WF \subset Q$ is the set of weakly forbidden states

We present in what follows the adaptive control algorithm. The first step is to explore the system FSM to construct the diagnoser FSM. The states of the diagnoser FSM are composed of possible system states according to observable actions and their failure status (N or F).
Once the diagnoser is built, it is possible to identify all failure states that will be reported to the controller. Then, the controller marks all failures as forbidden states. The next step is mark all states that lead to forbidden states by uncontrollable events as weakly forbidden states. Lastly, the controller acts on controllable events to avoid all forbidden and weakly forbidden states.

**Algorithm of Failure diagnosis with adaptive control**

Initially $Q^D = \{q_0^D\} = \{(q_0, N)\}$
Push $(q_0^D)$

While ($q^D$=pop stack) is not empty
  foreach $\sigma \in \Sigma_o$ do
    foreach $q$ from $q^D$
      if $(q, w\sigma w', q') \in \Delta$ such that $w \in \Sigma_{uo}^*$ and $w' \in \Sigma_{uo}^*$
      then generate/update state $q'^D$ with the appropriate status N or F
      endif
    end for
    push $q'^D$
    $Q^D = Q^D \cup \{q'^D\}$
    $\Delta^D = \Delta^D \cup \{(q^D, \sigma, q'^D)\}$
  end for
end while

foreach $q^D \in Q^D$ such that $q^D$ is Fi-certain
  mark each failure state q as forbidden
  $F = F \cup \{q\}$
end for

foreach $q \in F$ with $(q', w, q) \in \Delta$ and $w \in \Sigma_{uc}{}^*$
    mark $q'$ as weakly forbidden
        $WF = WF \cup \{q'\}$
end for

foreach $q \in F \cup WF$
  for each $(q', \sigma, q) \in \Delta$ such that $\sigma \in \Sigma_c$
    remove $(q', \sigma, q)$ from $\Delta$
  end for
end for

The aforementioned algorithm guarantees the safety specification properties. This is based on the diagnoser construction followed by adaptive control that benefits from diagnoser outputs. This Algorithm ensures that all forbidden and weakly forbidden states are unreachable by controller actions. In this case the adaptive control is successful.

Note that this process allows the controller to be maximum permissive since only forbidden states are removed and any state that reach forbidden states by uncontrollable events.

## 5. Conclusion

In this paper, we presented a method for adaptive control that consists of combining diagnose power with the controller to guarantee that the system remains as much as possible in safe state to guarantee reliability properties. The controller collaborates with the diagnoser in order to identify and detect failure states which are unobservable in most cases.

Future works focuses on the same problem in the context of real-time systems. Modelling is based on Timed Automata with some restriction to avoid the general undecidability of this problem when time is considered in dense context.

**Acknowledgments**

The authors wish to acknowledge the approval and the support of this research study by the grant N°CIT-2018-3-9-F-8037 from the Deanship of the Scientific Research in Northern Border University, Arar, KSA.

## References

[1] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D.Teneketzis, Diagnosability of discrete event systems, IEEE Transactions Automat. Contr.,vol. 40, no. 9,pp. 1555-1575, 1995.
[2] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, Failure diagnosis using discrete-event models, IEEE Transactions Automat. Contr., vol. 40, no. 2, pp. 105-124, 1996.
[3] F. Lin , Diagnosability of discrete-event systems and its applications, J. DEDS, vol. 4, no. 2, pp. 197-212, 1994.
[4] D. A. Pearce, The induction of fault diagnosis systems from qualitative reasoning, Proceedings of the AAAi National Conference on Artificial Intelligence, pp. 353-357, St Paul, Etats-Unis, 1988.
[5] Debouk, Rami, Stéphane Lafortune, and Demosthenis Teneketzis. "Coordinated decentralized protocols for failure diagnosis of discrete event systems." *Discrete Event Dynamic Systems* 10.1 (2000): 33-86.
[6] Lafortune, Stéphane, Feng Lin, and Christoforos N. Hadjicostis. "On the history of diagnosability and opacity in discrete event systems." *Annual Reviews in Control* 45 (2018): 257-266.
[7] Sasi, Yazeed, and Feng Lin. "Detectability of networked discrete event systems." *Discrete Event Dynamic Systems* 28.3 (2018): 449-470.
[8] Viana, Gustavo S., and João C. Basilio. "Codiagnosability of discrete event systems revisited: A new necessary and sufficient condition and its applications." *Automatica* 101 (2019): 354-364.

[9] Keroglou, Christoforos, and Christoforos N. Hadjicostis. "Distributed fault diagnosis in discrete event systems via set intersection refinements." *IEEE Transactions on Automatic Control* 63.10 (2018): 3601-3607.
[10] Wonham, W. M., Kai Cai, and Karen Rudie. "Supervisory control of discrete-event systems: A brief history." *Annual Reviews in Control* 45 (2018): 250-256.
[11] Yin, Xiang, and Zhaojian Li. "Decentralized fault prognosis of discrete-event systems using state-estimate-based protocols." *IEEE transactions on cybernetics* 49.4 (2018): 1302-1313.
[12] Wang, Deguang, Xi Wang, and Zhiwu Li. "State-based fault diagnosis of discrete-event systems with partially observable outputs." *Information Sciences* 529 (2020): 87-100.
[13] Alzalab, Ebrahim Ali, et al. "Fault-Recovery and Repair Modeling of Discrete Event Systems Using Petri Nets." *IEEE Access* 8 (2020): 170237-170247.
[14] Boussif, Abderraouf, Mohamed Ghazel, and Joao Carlos Basilio. "Intermittent fault diagnosability of discrete event systems: an overview of automaton-based approaches." *Discrete Event Dynamic Systems* 31.1 (2021): 59-102.
[15] Lamperti, Gianfranco, Marina Zanella, and Xiangfu Zhao. "Diagnosis of Deep Discrete-Event Systems." *Journal of Artificial Intelligence Research* 69 (2020): 1473-1532.
[16] Takai, Shigemasa. "A Generalized Diagnosability Condition for Diagnosis of Discrete Event Systems Subject to Sensor Failures." *IFAC-PapersOnLine* 53.4 (2020): 344-349.
[17] Takai, Shigemasa. "A general framework for diagnosis of discrete event systems subject to sensor failures." *Automatica* 129 (2021): 109669.
[18] Tan, Jianxin, et al. "Fault diagnosis of discrete-event systems under a general architecture." *Journal of Ambient Intelligence and Humanized Computing* (2021): 1-19.
[19] Alves, Marcos VS, et al. "Robust supervisory control of discrete event systems against intermittent loss of observations." *International Journal of Control* 94.7 (2021): 2008-2020.
[20] Ramadge, Peter JG, and W. Murray Wonham. "The control of discrete event systems." *Proceedings of the IEEE* 77.1 (1989): 81-98.
[21] Seatzu, Carla, Manuel Silva, and Jan H. Van Schuppen. *Control of discrete-event systems*. Vol. 433. Springer, 2013.
[22] Passino, K. M., and P. J. Antsaklis. "On the optimal control of discrete event systems." *Proceedings of the 28th IEEE Conference on Decision and Control,*. IEEE, 1989.
[23] Wonham, W. Murray, and Kai Cai. "Supervisory control of discrete-event systems." (2019): 2005-06.
[24] [24] Thistle, John G. "Supervisory control of discrete event systems." *Mathematical and Computer Modelling* 23.11-12 (1996): 25-53.
[25] Kumar, Ratnesh, and Vijay K. Garg. Modeling and control of logical discrete event systems. Vol. 300. Springer Science & Business Media, 2012.

**Yamen El Touati** received his Engineering, M.S, and PhD degrees In Computer Science from the National School of Computer Science (ENSI), University of Manouba in 2003, 2005 and 2014, respectively. He is an assistant professor at ISAMM, University of Manouba, Tunisia and a permanent research member of the OASIS laboratory at the National School of Engineers of Tunis, University of Tunis El Manar, Tunis, Tunisia. Currently, he is on secondment as assistant professor in computer science at the Department of Computer Science, Faculty of Computing and Information Technology – Northern Border University (NBU) in the Kingdom of Saudi Arabia. His research interests include modelling, diagnosis and control supervision of dynamic hybrid systems and timed systems with various automata based and Petri-net based models. His research also, focuses on security and opacity issues for composed web services.

**Mohamed Ayari r**eceived the Dipl.-Ing., M.S, and PhD degrees in Telecommunications respectively in 2003, 2004 and 2009 from the National Engineering School of Tunis (ENIT)-Tunisia in collaboration with National Polytechnic Institute of Toulouse-France and Virginia-Tech-USA. He is a teacher in several universities since 2003. His is a permanent research member in 6'COM laboratory at ENIT since 2003 till now. In 2005 he joined RCEM-Inc. at Toulouse-France. Since 2010 he has been a tenure track Assistant-Professor at National Engineering School of Carthage (ENICAR)-Carthage University-Tunisia. Since 2015 he is joined as assistant professor IT Department of Faculty of Computing and Information Technology – Northern Border University (NBU) in the Kingdom of Saudi Arabia. His current research interests are electromagnetic (EM) fields, numerical EM methods, computer-aided design of microwave circuits and antennas. His research interests include also information security and wireless applications..