Timed Residuals for Fault Detection and Isolation in Discrete Event Systems

Stefan Schneider, Lothar Litz, Mickaël Danancher

Timed Residuals for Fault Detection and Isolation in Discrete Event Systems

Stefan Schneider, Lothar Litz
Institute of Automatic Control
University of Kaiserslautern
Kaiserslautern, Germany
sschneider@eit.uni-kl.de, litz@eit.uni-kl.de

Mickaël Dananchер
LURPA
Ecole Normale Supérieure de Cachan
Cachan Cedex, France
mikael.dananchер@lurpa.ens-cachan.fr

Abstract—**In this paper a new attempt for fault detection and isolation in discrete event systems is proposed. An identified model constitutes a timed observer of the fault-free system behavior. Non-acceptable plant operation is detected by comparing the behavior of the model with the observed system output. For fault isolation, timed residuals and generic fault symptoms – early and late events – are introduced. Time bounds are composed using Boolean conditions and statistical analysis. In case of a fault, timed and untimed residuals are concluded in order to refine a set of potential faulty candidates. The method is applied to the given benchmark system of a virtual production plant with an external controller.**

Keywords-Discrete Event System; Timed Automata; Timed Residuals

I. INTRODUCTION

Fault Detection and Isolation (FDI) in industrial systems focuses on the reduction of production downtimes to increase availability. A particular challenge in this field is the development of diagnosis tools for large complex discrete event systems (DES). Several signal and model based approaches for different diagnosis applications have been introduced. Model based concepts perform a comparison of the modeled and the observed system behavior. In case of a deviation a fault is detected and isolated. The applied models can be characterized by two properties. First, models including faulty behavior and models which represent the fault-free behavior can be distinguished. The second property indicates whether the model includes time information or not. One example in literature is the diagnoser structure that models fault-free behavior as well as the behavior for given faults without considering time constraints. This class of models is studied in detail in [1] and an extension to dense-time automata is given in [2]. A Boolean decentralized structure with timed diagnosers is presented in [3] and an approach to timed FDI using fault-free models and template language is proposed in [4].

In this work timed residuals using timed fault-free models are introduced. Previous works provided an identification algorithm [5] to identify a monolithic automaton based on measured system data collected during fault-free system evolutions. An observer structure is used for fault detection purpose. Further developments presented a distributed approach [6] of untimed automata for DES. In order to reproduce concurrent system behavior the global system is divided into subsystems. Partial automata are identified to build a network with additional scalable restrictions on the behavior. A set of untimed residuals was introduced in [7] to perform fault isolation of the fault symptoms *unexpected* and *missed* behavior for a monolithic model. This enables a precise isolation of logic faults that occurred in the system. An extension of the fault-free model approach is presented in this paper considering new time based aspects. The fault symptoms *early* and *late* events are covered by timed residuals. Both timed and untimed residuals are treated as a compound.

The paper is structured as follows. In section 2 the timed model of a DES is introduced. A formal definition of the timed automaton model is given. The timed identification and composition of partial automata is explained. Section 3 deals with timed FDI. Time related faults are treated in detail including an illustrative example calculation. A case study of the benchmark system is given in section 4.

II. TIMED MODEL OF A DES

A. Problem classification

The observed system is considered as a closed-loop DES with information exchange between plant and controller. Since no knowledge about the control algorithm or plant structure is used the system is treated as a black box. Binary sensor signals are interpreted as controller inputs *I* and binary actuator signals as controller outputs *O*. Fig. 1 gives a schematic of the input/output (I/O) relation. In the following the controller is assumed fault-free, i.e. the controller software behaves deterministically. The physical system is non-deterministic because of temporal process variations. A DES state represents the combined state of controller and plant. FDI efforts are restricted to sensor and actuator faults of the plant.

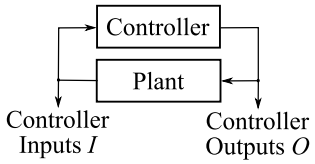

Figure 1. Closed-loop DES

B. Timed model definition

To discuss timed fault detection and isolation an appropriate formal model must be defined. According to the timed automaton with guards described in [8] an eight-tuple is introduced which is able to produce the same external behavior as the closed-loop DES with respect to time constraints. It is denoted as Timed Autonomous Automaton with Output (TAAO).

$$\text{TAAO} = (X, x_0, \Omega, \lambda, C, g, Tra, TG) \tag{1}$$

X : Finite set of states

x_0 : Initial state

Ω : Output alphabet

λ : Output function

C : Set of clocks

g : Clock mapping function

Tra : Set of timed transitions

TG : Set of time guards

X with cardinality $|X| = n$. Two successive states are denoted as x and x', $x, x' \in X$. $x_0 \in X$.

Ω with cardinality $|\Omega| = p$. Ω will be used to interpret the external behavior of the DES. The explanation is given in section timed model identification.

C with cardinality $|C| = |X|$ contains as many clocks as states.

g assigns a clock $c \in C$ to a state x

$$g : X \rightarrow C \tag{2}$$

where $g(x)$ addresses the clock c of state x. The mapping is bijective. A clock interpretation f is defined as

$$f : C \rightarrow \mathbb{R}^+ \tag{3}$$

where $f(c)$ represents the time value of clock c.

Tra is denoted as

$$Tra \subseteq X \times TG \times C \times X \ . \tag{4}$$

An element of the set is interpreted as

$$(x, tguard(x, x'), g(x'), x') \in Tra \ . \tag{5}$$

The element $g(x')$ represents the clock to be reset with this transition. Always the clock of the succeeding state x' is reset to zero, hence the transition labels are simplified and consist only of the time constraints $tguard(x, x')$.

TG contains Boolean conditions expressed as functions of clocks. A time guard $tguard \in TG$ is denoted as

$$tguard(x, x') = \left(f(g(x)) \geq \tau_{MIN}^{x,x'} \right) \wedge \left(f(g(x)) \leq \tau_{MAX}^{x,x'} \right). \tag{6}$$

$\tau_{MIN}^{x,x'}, \tau_{MAX}^{x,x'} \in \mathbb{R}^+$ constitute the time bounds of transition from x to x', $|TG| \leq n(n-1)$. $f(g(x))$ represents the value of the clock associated with x. An interval notation $f(g(x)) \in \left[\tau_{MIN}^{x,x'}, \tau_{MAX}^{x,x'} \right]$ of $tguard$ is used alternatively.

λ assigns an output $u \in \Omega$ to a model state x defined as

$$\lambda : X \rightarrow \Omega \, . \tag{7}$$

When x is activated $\lambda(x)$ ascertains the output of the TAAO.

Remark:

A distinction is drawn between logical and temporal non-determinism. A TAAO is logical deterministic if all guards are mutually exclusive out of a given state. A TAAO is always temporal non-deterministic since a transition may occur at any time within the defined time bounds.

C. Timed model identification

DES with large number of I/Os are basically able to exhibit a lot of different behavior patterns. Building a model by hand which is able to reproduce all system states is impracticable and usually even impossible. To resolve this difficulty an identification approach is chosen.

For FDI purposes the TAAO has to be identified. It is essential that the DES performs similar repetitive production cycles to obtain an appropriate data base. The data set contains the observed controller input and controller output sequences of the closed-loop DES in Fig. 1. They are called DES output sequences in the following. In this work we assume that the observed DES behavior is fault-free. Initially, the eight-tuple except TG is identified. The appropriate algorithm is available in [9]. Ω is used to accumulate the observed fault-free DES outputs. They are arranged in I/O vectors

$$u^{DES}(j) = \left(IO_1(j), \ldots, IO_m(j) \right) \tag{8}$$

with j-th event step and m number of controller I/Os, $|\Omega| \leq 2^m$.

The I/O enumeration convention is declared as follows. IO_i is defined as $IO_i = I_i \ \forall 1 \leq i \leq r$ with controller inputs $I_1, \ldots, I_r$ and $IO_{i+r} = O_i \ \forall 1 \leq i \leq s$ with controller outputs $O_1, \ldots, O_s$ and $m = r + s$ as defined in [9].

The concept for identification of TG is presented in the following. Based on all observed I/O vector sequences of the DES the corresponding time sequences are determined. With each new generated I/O vector a time span between two DES states is observed. This time span is called state sojourn time. With the identified TAAO so far and the determined state sojourn times a density interval distribution can be assigned to each transition. It shows how many times a modeled transition is observed within a defined time interval. Fig. 2 illustrates an example distribution of the benchmark system. Each transition in the automata model is related to one distribution. It may be noted that the obtained statistical data can be roughly approximated by a normal distribution. A statistical analysis leads to the determination of lower and upper time bounds for each transition in a generic way, e.g. $\mu - 3\sigma$ and $\mu + 3\sigma$ where μ denotes the mean value and σ the standard deviation. TG is based on the identified time bounds with $\tau_{MIN}^{x,x'}$ minimum and

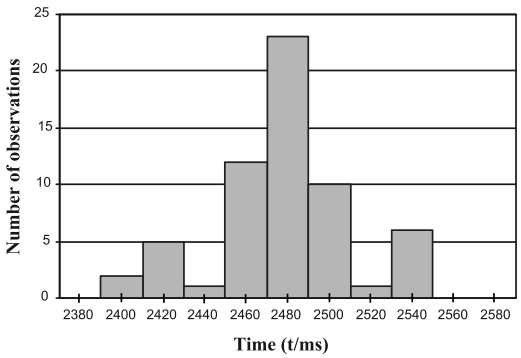

Figure 2. Measured density interval distribution of the benchmark system (interval length = 20ms)

$\tau_{MAX}^{x,x'}$ maximum time bound of all transitions. Multiple guards can apply the same bounds. The identified TAAO is able to reproduce the observed timed behavior of the closed-loop DES.

D. Distributed subsystems

For DES with a high degree of concurrency it is advantageous to identify subsystems based on sub-vectors of the I/O vector. A global model is composed of subsystem by grouping related sensor and actuator components. This partitioning operation is performed either using expert a-priori knowledge or automatically by means of optimization algorithms as treated in [6]. Partitioning is not a subject of this paper. The explanation of the timed residuals is restricted to the monolithic model approach. The extension to distributed models is straightforward.

III. Fault Detection and Isolation

A. Fault detection

FDI is performed based on the identified TAAO. Fig. 3 shows the basic online monitoring concept. The evaluator structure to observe the behavior of the DES is based on the timed system model. It is assumed that the current model state x is known. In case the DES generates a new event the model tries to reproduce the observed behavior. If the model contains a corresponding solution no deviation between the modeled and observed is concluded and the observed behavior is interpreted as acceptable. If the evaluator is not able to reproduce the observed behavior no succeeding state x' can be determined based on x and identified time bounds. Hence, a fault is declared.

The output of a DES is a sequence of timed events. Each event is described by a new I/O vector $u^{DES}(j)$, where $u^{DES}(j) \neq u^{DES}(j-1)$. To generate an event at least one I/O must change its value. The evolution of a single IO_i is denoted as single event e_{IOi}. Evolution Set ES contains all single events e_{IOi} between two I/O vectors $u(j)$ and $u(k)$

$$ES(u(j),u(k)) = \left\{ \begin{matrix} e_{IOi}\_1 \; if \; IO_i(j) = 0 \wedge IO_i(k) = 1 \\ e_{IOi}\_0 \; if \; IO_i(j) = 1 \wedge IO_i(k) = 0 \end{matrix} \right\} \quad (9)$$

$\forall 1 \le i \le m$ for the i-th vector element.

IO_i can change its value either from 1 to 0 denoted with e_{IOi}_0 or from 0 to 1 denoted with e_{IOi}_1. Since more than one I/O can change its value during an event step, the evolution set can be interpreted as the set of all occurred single events between two I/O vectors.

To accept an observed DES behavior, an active state x and a timed transition $(x, tguard(x,x'), g(x'), x')$, see (5), must exist which satisfy the logic condition

$$ES(\lambda(x), \lambda(x')) = ES(\lambda(x), u^{DES}(j)) \quad (10)$$

and the temporal condition

$$tguard(x,x') = true\,. \quad (11)$$

It is assumed that no fault was detected in the $(j-1)$-th step $\lambda(x) = u^{DES}(j-1)$. Fault detection is performed using the TAAO as fault-free system model. $\lambda(x)$ and $\lambda(x')$ determine the outputs of the model of x and x' according to definition (7). Based on the known x and all possible succeeding model states the logic condition checks whether the resulting ES is equal to the ES of the current model output $\lambda(x)$ and the new observed DES I/O vector $u^{DES}(j)$. If this holds for any succeeding state x', the TAAO can reproduce the observed DES behavior. In the next step the temporal condition has to be checked for all potential state candidates. At the time $u^{DES}(j)$ is observed the clock value $f(g(x))$ of x must be within the identified time bounds $\tau_{MIN}^{x,x'}$ and $\tau_{MAX}^{x,x'}$ according to the corresponding $tguard(x,x')$ between x and x'. If the condition is fulfilled the time guard returns *true*, otherwise *false*. A DES behavior is declared as acceptable, if both the logic and the temporal condition are met and an unambiguous succeeding state x' is determined.

A deadlock is concluded if no new I/O vector $u^{DES}(j)$ is observed based on x before the expiration of the maximum possible time bound of all potential transitions from x

$$\max_{\forall(x,tguard(x,x'),g(x'),x')} \left(\tau_{MAX}^{x,x'}\right). \quad (12)$$

In this section it is distinguished between logical and temporal misbehavior of the system and the fault detection is explained. If any of the described faults is detected the following fault isolation strategies are applied.

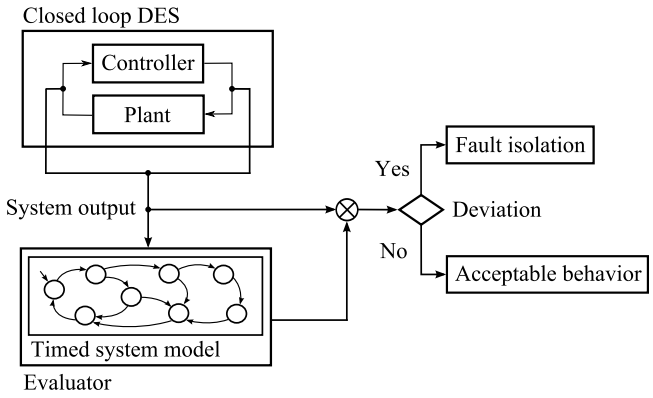

Figure 3. FDI conception

B. Residual approach

When a fault has been detected the goal is to isolate the fault by determining sensors and actuators which may cause the system to behave in a non-acceptable way. Since these hardware components are directly connected with the controller it is possible to determine faulty candidates by analyzing exclusively the controller I/Os. The residual approach of Roth introduced in [7] is an appropriate way to obtain a small number of I/Os which could be related to an observed logic fault. This work presents an attempt to formalize the deadlock symptom and the extension to timed residuals in order to handle the generic fault symptoms *early* and *late* events.

C. Deadlock isolation

The isolation strategy of a deadlock fault is based on the known residual *Res4* defined in [7]. It is again assumed that the current model state x is known and no new I/O vector $u^{DES}(j)$ is observed within the identified maximum possible time. *Res4'* is determined as

$$Res4'(x) = \bigcup_{\forall(x,tguard(x,x'),g(x'),x')} ES\left(\lambda(x),\lambda(x')\right). \quad (13)$$

The set $ES\left(\lambda(x),\lambda(x')\right)$ contains all single events e_{IOi} which would have led to a valid succeeding state x'. The union is applied to cover all possible states x' with an existing transition from x to x'. Since no behavior is observed any of the identified model transitions could be missed and hence all related single events could be the reason for the missing observation. *Res4'* denotes a special case of the *missed* behavior residual with $ES\left(\lambda(x),u^{DES}(j)\right)=\{\ \}$. It contains each missing single event which is possibly related to a deadlock fault.

D. Early and late behavior isolation

Faulty components can be isolated by determining behavior which is observed but *unexpected* or by *missed* events in a given context. In addition to these logical fault symptoms timed residuals *TRes*1 and *TRes*2 are introduced to deal with *early* and *late* events. A behavior which is observed out of time may be related to a faulty component.

The Time Guarded Evolution Set *TGES* contains all future and past single events between x and a succeeding state x'. It represents the modeled behavior which is expected to occur in the future or past with respect to the determined state sojourn time. The *TGES* is denoted as

$$TGES(x,x') = \left\{ES\left(\lambda(x),\lambda(x')\right) \mid \left(f\left(g(x)\right) < \tau_{MIN}^{x,x'}\right) \vee \left(f\left(g(x)\right) > \tau_{MAX}^{x,x'}\right)\right\} \quad (14)$$

with $f\left(g(x)\right)$ the state sojourn time of state x when $u^{DES}(j)$ is observed. Depending on $f\left(g(x)\right)$ a time attribute *early* or *late* is assigned to each *ES*.

$$\left(f\left(g(x)\right) < \tau_{MIN}^{x,x'}\right) \rightarrow early \quad (15)$$

$$\left(f\left(g(x)\right) > \tau_{MAX}^{x,x'}\right) \rightarrow late \quad (16)$$

If $\left(f\left(g(x)\right) < \tau_{MIN}^{x,x'}\right)$ holds, the actual observation $u^{DES}(j)$ occurred *before* the transition from x to x' may be taken due to the time bounds. In this case *TGES* contains the single events of the modeled fault-free system behavior marked by the label *early*.

If $\left(f\left(g(x)\right) > \tau_{MAX}^{x,x'}\right)$ holds, the actual observation $u^{DES}(j)$ occurred *after* the transition from x to x' may be taken due to the time bounds. In this case *TGES* contains the single events of the modeled fault-free system behavior marked by the label *late*.

The timed residual specification *TRes*1 represents a set of expected single events e_{IOi} which occurred *early* and *late* based on the current active model state x.

$$TRes1\left(x,u^{DES}(j)\right) = ES\left(\lambda(x),u^{DES}(j)\right) \cap \left(\bigcap_{\forall(x,tguard(x,x'),g(x'),x')} TGES(x,x')\right) \quad (17)$$

*TRes*1 is the intersection of the observed DES evolution $ES\left(\lambda(x),u^{DES}(j)\right)$ and all single events which are expected to occur in future or past no matter which following state is taken $\bigcap_{\forall(x,tguard(x,x'),g(x'),x')} TGES(x,x')$. The system evolution contains the single events between the output state x and the observed DES I/O vector $u^{DES}(j)$ which led to fault detection. The residual compares the behavior of the model with the actual observed system output. If the current observation of single events is equal to a behavior which should already have occurred (*late* behavior) or which has not yet been expected (*early* behavior) the according events are given by the residuals. It is also possible to give a less strict formulation using the union operation. This leads to the notation of the timed residual *TRes*2.

$$TRes2\left(x,u^{DES}(j)\right) = ES\left(\lambda(x),u^{DES}(j)\right) \cap \left(\bigcup_{(x,tguard(x,x'),g(x'),x')} TGES(x,x')\right) \quad (18)$$

with $TRes1 \subseteq TRes2$. *TRes*2 is usually less restrictive than *TRes*1 since it contains more elements. The results of both timed residuals are two small sets with possible faulty system components. In case of a fault the system operator should check the candidates of *TRes*1. If the fault cannot be found at these components the resulting elements of *TRes*2 should be considered to cover a wider field of potential candidates. In the following the attempt of timed residuals is applied to an illustrative example.

E. Illustrative example

A calculation is shown for the example TAAO in Fig. 4. It is assumed that x_1 is the current active state of the model and the new observed DES I/O vector is $u^{DES}(j)=(1,1)$ at time $f\left(g(x_1)\right)=5$. Since $f\left(g(x_1)\right)$ is smaller than the maximum upper time bound no deadlock occurred. The first step is to check whether the observed behavior can be reproduced by the model or not. Therefore the logical and temporal conditions for both successive states are applied.

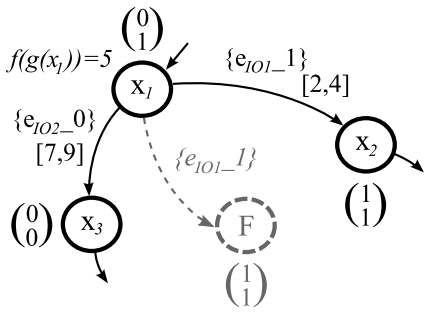

Figure 4. Example automaton

$$x_2 : ES\left(\lambda(x_1), \lambda(x_2)\right) = ES\left(\lambda(x_1), u^{DES}(j)\right) \quad (19)$$

$$x_3 : ES\left(\lambda(x_1), \lambda(x_3)\right) \neq ES\left(\lambda(x_1), u^{DES}(j)\right) \quad (20)$$

The logic condition does not hold for x_3. The list of potential successive states is thus reduced to x_2. The temporal condition yields

$$tguard(x_1, x_2) = false \quad (21)$$

since $f\left(g(x_1)\right) = 5$ is not within the time bounds of the considered transition between x_1 and x_2, a fault is detected. For fault isolation $ES\left(\lambda(x), u^{DES}(j)\right)$ is calculated to determine the evolution of the system.

$$ES\left(\lambda(x_1), u^{DES}(j)\right) = \{e_{IO1}\_1\} \quad (22)$$

Next, the modeled behavior is determined which is expected to occur in past or in future with respect to the measured clock $f\left(g(x_1)\right)$ of the active state x_1.

$$TGES(x_1, x_2) = \{e_{IO1}\_1\}, late \quad (23)$$

$$TGES(x_1, x_3) = \{e_{IO2}\_0\}, early \quad (24)$$

TGES of x_1 and x_2 contains the single event e_{IO1}_1 and the information that the occurrence would be *late* with respect to the state sojourn time. For the second transition the corresponding information is generated resulting in the single event e_{IO2}_0 and the *early* attribute. With the information about the observed and the modeled behavior it is possible to check whether the observed behavior is out of time and which single events have to be considered. *TRes*1 is determined as

$$\begin{aligned} TRes1 &= ES\left(\lambda(x_1), u^{DES}(j)\right) \cap \left(TGES(x_1, x_2) \cap TGES(x_1, x_3)\right) \\ &= \{e_{IO1}\_1\} \cap \left(\{e_{IO1}\_1\} \cap \{e_{IO2}\_0\}\right) = \{\ \} \end{aligned} \quad (25)$$

The calculation results in an empty set because of the fact that the transitions to both following states of the active state x_1 are characterized by different evolutions. Since no resulting candidates are obtained *TRes*2 is applied.

$$\begin{aligned} TRes2 &= ES\left(\lambda(x_1), u^{DES}(j)\right) \cap \left(TGES(x_1, x_2) \cup TGES(x_1, x_3)\right) \\ &= \{e_{IO1}\_1\} \cap \left(\{e_{IO1}\_1\} \cup \{e_{IO2}\_0\}\right) = \{e_{IO1}\_1\} \end{aligned} \quad (26)$$

*TRes*2 consists of the single event e_{IO1}_1. To obtain the temporal information the *TGES* is considered again and the entire result is determined as $\{e_{IO1}\_1\}, late$. This information is interpreted as IO_1 has changed its value from zero to one later than expected.

F. Residual interpretation

With the introduction of timed residuals another important class of fault symptoms is considered. It is shown logical and timed fault symptoms have to be distinguished. The presented fault detection approach is able to detect faults of both of the two domains. Timed residuals are an extension of the existing logical residuals. Hence, the fault isolation strategy must consider the combination of logical and timed residual calculation. A suitable scheme is illustrated in Fig. 5. Two dimensions of fault isolation are shown. *Unexpected* and *missed* behavior symptoms constitute the logical dimension. The timed residuals investigating the expected behavior which occurs *early* or *late* represent the timed dimension.

IV. CASE STUDY

The proposed benchmark system (Fig. 6) is the virtual pick and place station of the ITS PLC simulation environment for industrial systems. Running on a PC it is connected with a real programmable logic controller (PLC) via a data acquisition box to build a virtual automated manufacturing system. Multiple virtual sensors S and actuators A are available to control the system. Each component is labeled based on its type and an individual number.

In the following the specification of the system structure and production process for this work is outlined. Conveyor (S2, A0) provides parts and conveyor (S3, A1) empty boxes to the corresponding pick and place station. The gripper is moved horizontally between the two conveyors by two double acting pneumatic cylinders (S4, S5, A2, A3, A4, A5) and vertically using the single acting pneumatic cylinder (S6, S7, A6). Parcels are placed inside a box using the magnetic gripper (S8, A7). The detailed description of the benchmark system refers to [10]. A production cycle is specified as the filling of one box with nine parts. Initially, one box and a sequence of parts are transported. Parts are sorted consecutively into the box without consideration about the type of each part. The fully loaded box is then delivered to the exit conveyor belt.

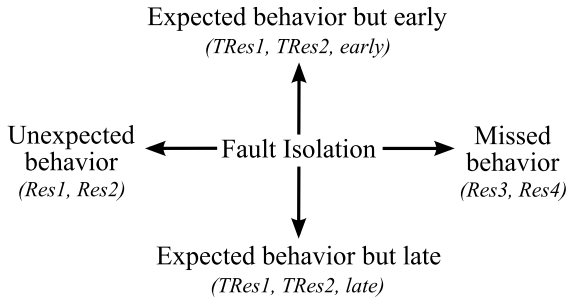

Figure 5. Fault isolation dimensions

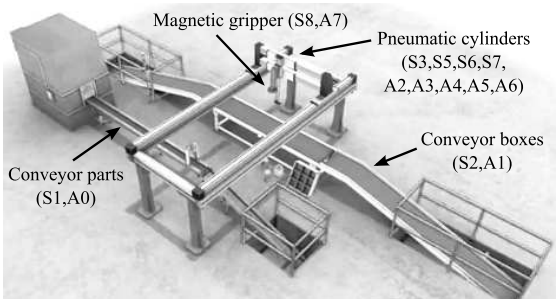

Figure 6. Pick and plance benchmark system

The system model is composed of distributed partial automata. Four TAAOs are identified, one for each of the two horizontal cylinders, one for the vertical cylinder including the gripper and one for the conveyors. The identification data is based on twenty production cycles of fault-free system evolutions. Time bounds are generated according to the presented attempt using μ and 3σ.

The simulated system enables introducing a variety of faults into the production system. In the following the investigated faults are related to the vertical operating cylinder exclusively. Three different faults are simulated:

Fault #1: After sorting the first parcel into the box, the extended vertical cylinder pulls back. Its arrival in the upper position is indicated by sensor S6. The sensor value is supposed to switch from zero to one as soon as the cylinder is completely contracted. It is assumed that sensor S6 is faulty. It is forced to switch *early* before the cylinder reaches its initial position.

Fault #2: The situation is the same as described with fault #1. S6 is assumed to be faulty again. In this case the sensor is forced to switch *late* from zero to one.

Fault #3: A stuck open fault of actuator A7 after the first parcel has been sorted represents to the third fault case. When the second parcel arrives at the pick station, the vertical cylinder is located in its upper initial position. Afterwards, the cylinder is supposed to move down in order to grip the object. S6 switches to zero and the controller awaits the response of S7, reporting the complete extension of the cylinder. It is assumed that A7 is faulty. It is not able to start working and the plant remains in a deadlock state.

The FDI results are summarized in Table 1. With the presented method it is possible to detect all given faults in real time. Only very few false alarms are generated due to properly chosen time bounds. Fault #1 is related to the generic fault symptom *early* event. No deadlock has occurred, hence set *Res4'* remains empty. *TRes1* and *TRes*2 contain the same result as all outgoing transitions of the active state have at least e_{S6}_1 as mutual single event. The sets return S6 as a potential faulty component which showed *early* behavior. The timed residuals of fault #2 are listed in the second row. In this case S6 is isolated as well as the potential faulty component since the corresponding event is observed *late.* A deadlock is simulated by inducing fault #3. The event of the cylinder extension is not observed within the maximum possible time bound. Therefore *Res4'* is calculated to determine the single events which may be related to the fault. One can recognize that the faulty component A7 is represented in the residual set by the e_{A7}_1. By applying the method of timed residuals all faulty components are isolated accurately.

TABLE I. EXPERIMENTAL RESULTS

Fault	**Residuals**		
	Res4'	***TRes1***	***TRes2***
Fault #1	{ }	$\{e_{S6}\_1\}, early$	$\{e_{S6}\_1\}, early$
Fault #2	{ }	$\{e_{S6}\_1\}, late$	$\{e_{S6}\_1\}, late$
Fault #3	$\{e_{S7}\_1,$ $e_{A7}\_1\}$	{ }	{ }

V. CONCLUSIONS

Timed residuals and timed fault detection is presented as a new attempt for FDI in DES. A timed automaton model denoted as TAAO is introduced. Fault detection is performed based on identified logic and timed conditions. The generic fault symptoms deadlock as well as *early* and *late* behavior are treated to isolate a small set of potential faulty candidates. The ability of the method is demonstrated by means of the given benchmark system.

REFERENCES

[1] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, "Diagnosability of Discrete-Event Systems," IEEE Transactions on Automatic Control, vol. 40, no. 9, pp. 1555-1575, September 1995.

[2] S. Tripakis, "Fault Diagnosis for Timed Automata," Proceedings of the 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems, LNCS 2469, pp. 205-221, September 2002.

[3] M. Sayed Mouchaweh, A. Philippot and V. Carré-Ménétrier, "Decentralized diagnosis based on Boolean discrete event models: application on manufacturing systems," International Journal of Production Research, vol. 46, iss. 19, pp. 5469-5490, October 2008.

[4] D. Pandalai and L. Holloway, "Template Languages for Fault Monitoring of timed Discrete Event Systems," IEEE Transactions on Automatic Control, vol. 45, no. 5, pp. 868-882, March 2000.

[5] S. Klein, J.-J. Lesage, L. Litz, "Fault detection of Discrete Event Systems using an identification approach," 16th IFAC World Congress, Praha(CZ), July 2005.

[6] M. Roth, J.-J. Lesage and L. Litz, "Black-box identification of discrete event systems with optimal partitioning of concurrent subsystems," Proceedings of the 2010 American Control Conference, pp. 2601-2606, June 2010.

[7] M. Roth, J.-J. Lesage and L. Litz, "A residual inspired approach for fault localization in DES," 2nd IFAC Workshop on Dependable Control of Discrete Systems (DCDS'09), November 2009.

[8] Cassandras, C. G. and Lafortune, S. Introduction to Discrete Event Systems, 2nd ed., Springer Verlag, 2008.

[9] M. Roth, "Identification and fault diagnosis of industrial closed-loop discrete event systems," Ph.D. thesis, Logos Verlag, 2010.

[10] A. Philippot, "Survey on diagnosis of a pick and place benchmark," Proceedings of the 3rd International Workshop on Dependable Control of Discrete Systems (DCDS'11), 2011.