Journal of Computers Vol. 31 No. 1, 2020, pp. 18-33
doi:10.3966/199115992020023101002

HMM-based Fault Diagnosis for Web Service Composition

Zhi-Chun Jia[1*], Yuan Lu[1], Xiang Li[1], Xing Xing[1]

[1] College of Information Science and Technology, Bohai University, Jinzhou, Liaoning, China
zhichun.jia@bhu.edu.cn

Received 3 April 2018; Revised 25 October 2018; Accepted 18 November 2018

Abstract. To reduce the time required in developing the web services fault tolerant, researchers have investigated some methods on automating the process of fault diagnosis. Most existing techniques either assume a complete process model or fault types available in the diagnostic system. Moreover, empirical studies have shown that the existing diagnosis methods are not adequate to meet the growing requirements of business processes. In this paper, we propose a Hidden Markov Model (HMM) based diagnosis method for autonomously diagnosing the faults in web service composition process. Our diagnosis method incorporates historical process data into model-based diagnosis for overcoming the limitations of the uncompleted process model and finite historical process data. Our diagnosis result can provide an explanation for the service faults. Moreover, we present an architecture of diagnostic system for simplifying diagnostic process, adding diagnostic capability and guaranteeing the privacy of web services. The experimental results show that our method is effective and robust to various noises in diagnosing the faults of web services.

Keywords: fault diagnosis, Hidden Markov Model, historical data, web service

1 Introduction

The increasing size and complexity of web service composition process has led to an amplified number of potential failures. These failures make the application process harder to ensure service reliability. To detect and recover from the potential failures, appropriate fault diagnosis techniques have been developed for web services (e.g. [1-2]), especially the diagnosis methods to detect and isolate faults during service execution in an efficient and effective way are desirable [3].

The fault diagnosis focuses on diagnosing faulty causes so that the business process is back to a normal, safe, operating state [4-5]. So far two categories of fault diagnosis techniques have been proposed. The Model-based Diagnosis methods (MBD) require the knowledge about structure, function and behavior of services to infer the cause of a failure. The recent historical data-based methods work on the statistical model of web services, which builds on historical execution information of web services [6-9].

The MBD methods provide the device-independent diagnostic reasoning procedures to determine whose incorrect behavior explains a given set of observations in the component services [10]. Current methods for diagnosing the faults during the composite service execution either assume the existence of an independent formal specification [3, 11-13] or of the explicit fault models for the service composition process [14-18]. Moreover, Petri nets [19] and automata [12, 20] have been used for modeling the web service process. Using the formal models and the runtime observations from the execution of business processes, these methods can identify the faulty behaviors and explain the faulty causes. Unfortunately, detailed specification and fault models for the service processes aren't necessarily generally be satisfied in practice. As the method is hard to fully capture the semantics of web services, it can't always correctly diagnose the faults during the composite service execution [21].

In contrast to MBD, historical data-based methods only consider the historical process data [22]. Different techniques to convert historical data into a diagnosis model as a priori knowledge can be

[*] Corresponding Author

applied for the historical data-based methods [23-24]. For example, Bayesian network technology [25], the technology based on statistical probability [26-28], and the neural network technology [29] and so on. The historical data-based methods are easy to implement and able to handle noise effectively. However, the effectiveness of these methods is limited by the availability of samples of historical data. Moreover, most of these methods don't provide an explanation for the faulty causes of the web service composition process.

In a word, MBD presumes the complete knowledge about the behaviors of web services while statistical methods are easy to set up but are vulnerable to semantic faults. Hence, they are not adequate to meet the growing requirements of business processes for a diagnostic system. Integrating these complementary features is one way to develop the hybrid methods. It could overcome the limitations of individual solution strategies [22].

Along with this line, our work focuses on the web services. We propose a HMM-based hybrid diagnosis method for diagnosing service composition faults. Computing the correct execution sequence with the maximum likelihood, our method compares the correct one with the exception execution sequence to localize the service faults. The main contributions of this paper include the followings:

(1) We present a diagnostic architecture for simplifying the diagnostic process, improving the diagnostic capability and guaranteeing the privacy of web services, where a local diagnosis service is associated to a BPEL service. And all local diagnosis services cooperate with a global diagnosis service.

(2) We build a HMM-based diagnosis model which not only exploits process description but also uses historical execution information to model.

(3) We propose a HMM-based fault diagnosis method for diagnosing the faults in web service composition processes. Our method incorporates historical process data into model-based diagnosis. Moreover, our method can provide an explanation for the faulty causes (by the correct execution trace).

(4) We provide a case study of BPEL process for business travel planning to evaluate the presented method. Experimental results show that our method is effective and robust to various noises in diagnosing the faults of web services.

The rest of paper is organized as follows. A framework of diagnosis system is given in Section 2. Section 3 introduces the related definitions and presents HMM-based diagnosis model and three algorithms. Section 4 discusses our experimental results. Section 5 briefly discusses related works. Finally, Section 6 gives the conclusion.

2 Architecture of Diagnosis System

In developing the fault tolerant of web services, exception handling techniques have been adopted in automatic web service choreography to support a graceful termination. The recovery actions depend on the observable effects of the occurred problems, rather than their causes [30-31]. A diagnostic system is thus proposed to enable more effective recovery strategies towards completion by detecting the faulty behaviors and recognizing the faulty causes of the thrown exceptions quickly and exactly.

For a business process, it can run down for many reasons. We mainly consider functional and behavioral faults such as value mismatch, missing data or parts of a message, wrong order of operation invocations and so on. For facilitating diagnosis, our diagnostic system assumes that either a BPEL (business process execution language) process of web service composition is known or their historical execution data are kept in service records such that:

(1) executed behaviors, including their inputs and outputs in the SOAP (simple object access protocol) messages, are completely recorded in a sequence way;

(2) service records are either successful execution traces of a BPEL process or failing traces including thrown exceptions and exception events, a service record $sr = \{(in_1, b_1, out_1), ..., (in_n, b_n, out_n), er\}$, where in_i ($1 \le i \le n$) denotes the input of a behavior, out_i ($1 \le i \le n$) denotes the output of a behavior, *er* denotes the execution result (*success* or *fail*) of the service composition process.

On the basis of the above assumptions, we propose multi-layer diagnosis architecture in Fig. 1. Our framework of diagnostic system mainly includes the following parts:

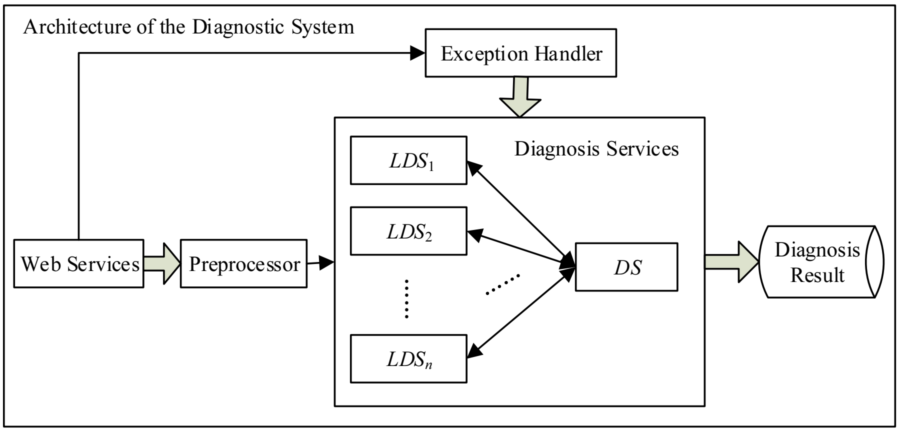

Fig. 1. Architecture of diagnosis system

—We extract BPEL files and service records from web service composition process.

—The preprocessor first models BPEL processes using Petri nets. Then it computes transition and emission probabilities according to process models and service records for modeling diagnosis model.

—The exception handler is responsible for catching the thrown exceptions and triggering diagnosis services.

—We associate each BPEL process with a local diagnosis service $\boldsymbol{LDS_i}$ $(1 \le i \le n)$. The BPEL processes include web service composition process and its sub-processes. The role of local diagnosis services is to diagnose the corresponding services and provide the global diagnosis service ***DS*** with the diagnostic results.

—The ***DS*** is responsible for passing the diagnostic information between the local diagnosis services, checking the diagnostic states for each service, ending diagnosis process and saving diagnostic results to the database.

—The database of diagnostic result is used to store diagnostic results including faulty behaviors and the correct execution trace.

The entire diagnostic interaction process is described as following:

—When an exception occurs in a process of web service composition, the exception handler triggers the corresponding local diagnosis service $\boldsymbol{LDS_k}$ $(1 \le k \le n)$.

—According to input and output messages of observation and our diagnosis model, the $\boldsymbol{LDS_k}$ firstly diagnoses the data faults of input and output messages and gives a correct explanation – a variable pair sequence.

—If faulty behavior invokes another web service $\boldsymbol{LDS_j}$, the $\boldsymbol{LDS_k}$ informs the ***DS*** of diagnosing the $\boldsymbol{LDS_j}$ and provides the fault messages to ***DS***.

—Then the local diagnosis service applies the given diagnosis algorithm to find the correct behavior sequence of maximum likelihood with the correct variable pair sequence. Comparing the correct behavior sequence with actual observed behavior sequence, local diagnosis service finds differences between the two behavior sequences. These differences could be faults about a wrong order of operation invocations.

At last, the local diagnosis service sends the diagnostic results to the global diagnosis service. When no web services need to be diagnosed or all services have been diagnosed, the global diagnosis service ends diagnosis process and stores diagnostic results to the database.

In our architecture, it is important to notice that we consider a general case where:

—The component services can be added, removed and replaced independently at any time. Such a decoupling makes individual diagnostic process be simplified.

—Each local diagnosis service can adopt its own diagnostic strategy according to the availability of the corresponding process model and service record. It adds the diagnostic capability of our diagnostic system in a complex business process.

—The local diagnosis services only communicate with the global diagnosis. They don’t know each other, making it possible to guarantee the privacy of web services.

3 Modeling BPEL Process for HMM-based Diagnosis

Our framework uses HMM to model BPEL processes for fault diagnosis of web services, and incorporates service records into our HMM-based diagnosis model. In this section, we briefly recall the HMM before describing the proposed diagnosis model and algorithms.

3.1 HMM and Viterbi Algorithm

3.1.1 Hidden Markov Model

The Hidden Markov Model is a finite set of states, each of which is associated with a probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are “hidden” to the outside; hence the name Hidden Markov Model.

Definition 1. A Hidden Markov Model is 5-tuple, $\lambda = (X, O, \pi, A, B)$ where:

— A finite set of states in the model: $X = \{s_1, s_2, \cdots, s_n\}$, where *n* is number of states;

— A set of observation symbols: $O = \{v_1, v_2, \cdots, v_m\}$, where *m* is number of observation symbols;

— Initial state distribution: $\pi = \{\pi_i, 1 \le i \le n\}, \pi_i = P(X_1 = s_i)$, where X_1 denotes initial state;

— $A = \{a_{ij}, 1 \le i, j \le n\}, a_{ij} = P(X_{t+1} = s_j \mid X_t = s_i)$ is a set of state transition probabilities, where X_t denotes the current state. Transition probabilities should satisfy the normal stochastic constraints $a_{ij} > 0$ and $\sum_{j=1}^{n} a_{ij} = 1$;

— $B = \{b_{ik}, 1 \le i \le n, 1 \le k \le m\}, b_{ik} = P(O_t = v_k \mid X_t = s_i)$ is a emission probability distribution in each of the states, where v_k denotes the *k*th observation symbol and O_t denotes the current observation.

3.1.2 Assumptions in HMM

For the sake of mathematical and computational tractability, following assumptions are made in the theory of HMM:

— **Markov Assumption:** It is assumed that the next state is dependent only upon the current state;

— **Stationarity Assumption:** It is assumed that state transition probability is independent of the actual time at which the transition takes place;

— **Output Independence:** This is the assumption that current output is statistically independent of the previous output;

3.1.3 Viterbi Algorithm

There are three basic problems of interest in HMM. They are evaluation problem, decoding problem and learning problem. Evaluation problem can be used for isolated recognition. Decoding problem is related to the continuous recognition as well as to the segmentation. Learning problem must be solved, if we want to train an HMM for the subsequent use of recognition tasks.

In order to find the most likely state sequence for a given sequence of observations $\sigma = \{O_1, O_2, \cdots, O_T\}$ and a HMM model, we use Viterbi algorithm to find the entire state sequence with maximum likelihood. Viterbi algorithm is ascribed below:

— **Initialization**

$\delta_1(s_i) = \pi_i b_i(O_1)$, $\phi_1(s_i) = 0$, $1 \le i \le n$;

— **Recursion**

$\delta_t(s_j) = \max_{1 \le i \le n}[\delta_{t-1}(s_i) a_{ij}] b_j(O_t)$, $\phi_t(s_j) = \arg\max_{1 \le i \le n}[\delta_{t-1}(s_i) a_{ij}]$, $2 \le t \le T$, $1 \le j \le n$;

— **Stop**

$P^* = \max_{1 \le i \le n}[\delta_T(s_i)]$, $q_T^* = \arg\max_{1 \le i \le n}[\delta_T(s_i)]$;

— **State sequence with maximum likelihood**

$q_t^* = \phi_{t+1}(q_{t+1}^*)$, $t = T-1, T-2, \cdots, 1$;

— **Time complexity**

$O(n^2 T)$.

3.2 HMM-based Diagnosis Model

The HMM-based diagnosis model extends the BPEL process models with service records. For modeling the fault diagnosis model, we firstly need to compute transition and emission probabilities for the behaviors and variables in BPEL process models and service records. Then we combine the transition probability and emission probability of process model and service records according to their weights.

3.2.1 Model Definition

Definition 2. A HMM-based Diagnosis Model is $HDM = (X, V, O, \pi, A, B, VA)$, where:

— The set of behaviors: $X = \{t_i, 1 \le i \le n\}$, where *n* denotes the number of behaviors in the process model;

— The set of variables: $V = \{v_i, 1 \le i \le l\}$, where *l* denotes the number of variables in the process model;

— $Wf \subseteq (X \times V) \bigcup (V \times X)$ is the workflow relation in composite web service; If $(v,t) \in Wf$ for a behavior *t* and a variable *v*, then *v* is an input of *t*; If $(t,v) \in Wf$ for a behavior *t* and a variable *v*, then *v* is an output of *t*; Let $x, y \in X \bigcup V$, the set ${}^\bullet y = \{x \mid (x,y) \in Wf\}$ is called the pre-set of *y* and the set $x^\bullet = \{y \mid (x,y) \in Wf\}$ is the post-set of *x*;

— The set of variable pair: $O = \{vp_k = (v_i, v_j),\ 1 \le k \le m, 1 \le i, j \le l\}$, where *m* denotes the number of variable pairs in the process model;

— Initial behavior distribution: $\pi = \{\pi_i = 1, \pi_j = 0 \mid\ {}^\bullet t_i = p_s, j \ne i, 1 \le i, j \le n\}$, where p_s denotes start place of the process model;

— The set of behavior transition probability: $A = \{a_{ij}, 1 \le i, j \le n\}$, $a_{ij} = P(X_{t+1} = t_j \mid X_t = t_i)$;

— The set of emission probability: $B = \{b_{ik}, 1 \le i \le n, 1 \le k \le m\}$, $b_{ik} = P(O_t = vp_k \mid\ X_t = t_i)$;

— The set of variable transition probability: $VA = \{va_{ij}, 1 \le i, j \le l\}$, $va_{ij} = P(X_{t+1} = v_j \mid X_t =\ v_i)$.

In the BPEL process model, all behaviors are equally likely to happen. Thus, all transition probabilities starting from the same behavior are equal. And all transition probabilities starting from the same variable are equal and all emission probabilities starting from the same behavior are equal.

Definition 3. A Model Behavior Transition Probability is $MA = \{ma_{ij}, 1 \le i, j \le n\}$, where:

— If $t_i^\bullet = {}^\bullet t_j$ then $p_{ij} = 1$, otherwise $p_{ij} = 0$;

— $ma_{ij} = p_{ij} / \sum_{j=1}^{n} p_{ij}$;

— Here, there is a special structure – ‘flow’ in the BPEL process. The order of behaviors in the flow structure is not unique. We need to consider all possible orders in our diagnosis model.

Definition 4. A Model Emission Probability between behavior and variable pair is $MB = \{mb_{ik}, 1 \le i \le n, 1 \le k, r \le m\}$, where:

— If $({}^\bullet t_i).v = v_i$ and $(t_i^\bullet).v = v_j$, then $p_{ik} = 1$, otherwise $p_{ik} = 0$, where $({}^\bullet t_i).v$ denotes the variable in the input place of t_i , the same for $(t_i^\bullet).v$;

— $mb_{ik} = p_{ik} / \sum_{k=1}^{m} p_{ik}$.

Definition 5. A Model Variable Transition Probability is $MV = \{mva_{ij}, 1 \le i, j \le l\}$, where:

— if $p_i^\bullet = {}^\bullet p_j$ and $p_i.v = v_i, p_j.v = v_j$, then $p_{ij} = 1$, otherwise $p_{ij} = 0$, where $p_i.v$ denotes the variable of place p_i ;

— $mva_{ij} = p_{ij} / \sum_{j=1}^{l} p_{ij}$.

Definition 6. A Record Behavior Transition Probability is $RA = \{ra_{ij}, 1 \le i, j \le n\}$, where:

— n_i denotes the number of t_i in the service records;

— n_{ij} denotes the number of that t_j is the next behavior of t_i in the service records;

— $ra_{ij} = n_{ij} / n_i$.

Definition 7. A Record Emission Probability is $RE = \{rb_{ik}, 1 \le i \le n, 1 \le k \le m\}$, where:

— n_i denotes the number of t_i in the service records;

— vn_{ik} denotes the number of that the input of t_i is $vp_k.pre$ and the output is $vp_k.next$ in the service records;

— $rb_{ik} = vn_{ik} / n_i$.

Definition 8. A Record Variable Transition Probability is $RV = \{rva_{ij}, 1 \le i, j \le l\}$, where:

— vn_i denotes the number of variable v_i in the service records;

— vvn_{ij} denotes the number of that v_j is the input variable of *t* and v_i is the output variable of *t* in the service records;

— $rva_{ij} = vvn_{ij} / vn_i$.

Definition 9. A HMM-based Behavior Transition Probability is $A = \{a_{ij}, 1 \le i, j \le n\}$, where:

— θ_m is the weight of the process model;

— θ_r is the weight of the service records;

— $\theta_m + \theta_r = 1$ and $a_{ij} = \theta_m ma_{ij} + \theta_r ra_{ij}$.

Definition 10. A HMM-based Emission Probability is $B = \{b_{ik}, 1 \le i \le n, 1 \le k \le m\}$, where:

— θ_m is the weight of the process model;

— θ_r is the weight of the service records;

— $\theta_m + \theta_r = 1$ and $b_{ik} = \theta_m mb_{ik} + \theta_r rb_{ik}$.

Definition 11. A HMM-based Variable Transition Probability is $VA = \{va_{ij}, 1 \le i, j \le l\}$, where:

— θ_m is the weight of the process model;

— θ_r is the weight of the service records;

— $\theta_m + \theta_r = 1$ and $va_{ij} = \theta_m mva_{ij} + \theta_r rva_{ij}$.

How to select the parameters θ_m and θ_r ? When the service records are finite, θ_m should be greater than θ_r. When the service records are enough and process model is not complete or explicit, θ_r should be greater than θ_m .

Definition 12. A Sequence of Observations is $\sigma = \{O_1, O_2, \cdots, O_T\}$, where:

— *T* denotes the number of observation behaviors (variable pairs) in σ ;

— O_t includes the observation variables (including input and output variables) and behavior *t*.

3.2.2 Algorithm Description

Based on the above-mentioned HMM-based diagnosis model, our method focuses on diagnosing the faults of composition service itself, such as business logic faults and data semantics faults. These faults are the most difficult to identify but the most critical for composition applications.

Our diagnosis goal is to find a correct behavior transition and localize their differences as faults. In here, the correct behavior transition denotes the best matching execution trace with the exception execution.

Firstly, we make the following assumptions:

— Either BPEL process description or historical execution data is available;

—We are able to convert process description or historical data into our diagnosis model;

—The exception execution information is available, including executed behavior and variable sequence.

Second, the **algorithm *VD*** is applied to checking every variable transition in the exception execution. If the probability of a variable transition is less than the **threshold** Δ, we think it is a faulty variable transition. Finally, the **algorithm *VD*** can find a correct variable transition to substitute the faulty one, and obtain a correct variable sequence.

Third, we use the **algorithm *BOD*** find a correct behavior sequence with the maximum likelihood by the correct variable sequence.

Finally, the **algorithm *BOD*** compares the correct sequences with the exception execution, and locates the discrepancies between them as the service faults.

```
Algorithm. VD (HDM, σ, VQ, VS)
Input: HDM - diagnosis model; σ - observation sequence.
Output: VQ - the correct variable pair sequence;
        VS - diagnosis solution of variables.
VS = null; VQ = variable pair sequence in σ;
for i = 1; i <= VQ.lenght; i++;
    vp = variable pair in VQ(i);
    vp.in = input variable of vp;
    vp.out = output variable of vp;
    if HDM.VA(vp) < Δ;
    then
       VS = VS⋃σ(i);
        if vp.in doesn't exist or
               all values starting from vp.in in HDM.VA are 0;
        then VQ(i) = max(HDM.B(max(HDM.A(σ(i-1))));
        else VQ(i) = max(HDM.VA(vp.in));
        end if
    end if
end for
return (VQ, VS);
```

In the **algorithm *VD***, the *max*() returns a variable pair or behavior transition, and the probability of the variable pair or the behavior transition is the maximum probability starting from the given variable or behavior in the given the probability set. If the transition from *vp.in* to *vp.out* is faulty, the corresponding transition probability *HDM.VA*(*vp.in*, *vp.out*) is far less the normal transition probability. So, we set the threshold value to $\Delta = 1/2nz_i$. In here, nz_i ($nz_i \neq 0$) denotes the number of values in *HDM.VA*(v_i). If the probability *HDM.VA*(*vp*) of the variable pair *vp* is less than Δ, we think the variable pair *vp* is a faulty variable pair, and put *vp*, its input and output into the diagnosis solution *VS*. Then we find the correct output variable instead of the output variable of *vp*. Finally, the **algorithm *VD*** gets a correct variable sequence *VQ* and a diagnosis solution of variables *VS*. The **time complexity** of the **algorithm *VD*** is $O(Tl)$. The l denotes the number of variables in the process model. And the T denotes the number of observation behaviors (variable pairs) in the observation sequence.

```
Algorithm. BOD (HMD, σ, VQ, VS, LS)
Input: HMD, σ, VQ, VS.
Output: LS-a local diagnosis solution.
LS = VS; HDM' = HDM;
if the probabilities in HDM' are 0;
then the probabilities in HDM' is ε;
end if
sq = Viterbi(HMD', VQ);
for i=1; i <= sq.length
  if sq(i)!= σ(i).behavior;
```

```
08.   then LS = LS⋃σ(i).behavior;
09.   end if
10. end for
11. return LS;
```

According to the correct variable sequence *VQ* and the diagnosis solution of variables *VS* from the **algorithm *VD***, the **algorithm *BOD*** can diagnose the faulty behaviors and the wrong order of operation invocations in a local diagnosis service. The **algorithm *BOD*** first checks all probabilities in the diagnosis model *HDM*. If the probability of a transition is zero, then the algorithm changes it into the given minimum value ε . The ε is far less than the other probabilities in *HDM*. After that, the algorithm gets a new diagnosis model *HDM'*. Next, we use Viterbi algorithm to find the entire behavior sequence *sq* with maximum likelihood according to the *VQ* and *HDM'*. We compare the *sq* with the behavior sequence in σ, and find the different behaviors in the same position. We think that the different behaviors in the two sequences are faulty behaviors. The algorithm puts the faulty behaviors into the local diagnosis solution *LS*. Finally, the **algorithm *BOD*** gets a complete local diagnosis solution *LS*. The time complexity of the **algorithm *BOD*** is $O(n^2T)$, where the *n* denotes *t* the number of behaviors in the process model and the *T* denotes the number of observation behaviors (variable pairs) in the observation sequence.

4 Experiment

To validate the effectiveness of our method, we first set up a simulation environment for generating a large amount of experimental data and achieving the diagnostic results. Moreover, we conduct a case study to illustrate the effectiveness of our method. Finally, we evaluate our method in our simulation environment.

4.1 Simulation Environment

To evaluate our method, we design a simulation experimental environment to automatically generate workflows and service execution logs according to the given parameters. Our simulation experimental environment shown in Fig. 2 consists of two parts **Generator** and **Diagnoser**. The **Generator** can generate the service workflows and their running logs. In the **Generator**, the **DF** is responsible for recording the information of the web services. The **CA** is used to build the service workflow by reading the **BPEL** file. It selects the suitable service components from the **DF** to build the composite web service, and generates the service running log file **SEL**. The **Diagnoser** can diagnosis the service faults and explain the faulty reason. In the **Diagnoser**, the **model converter** is responsible for converting **BPEL** file into the HMM-based model. The **data converter** is in charge of converting the **SEL** into the HMM-based model. The **combiner** combines the two models into the composite HMM-based diagnosis model. The **noise box** is used to add the noise data in the models. We define three different types of noise generating operations: (1) to delete a part of the event sequence; (2) to interchange two random chosen behaviors (or variable pairs); (3) to use an unknown behavior (or variable) instead of a behavior (or variable) in the event sequence. The **diagnosis service** is responsible for diagnosing the process faults and generating the diagnosis result.

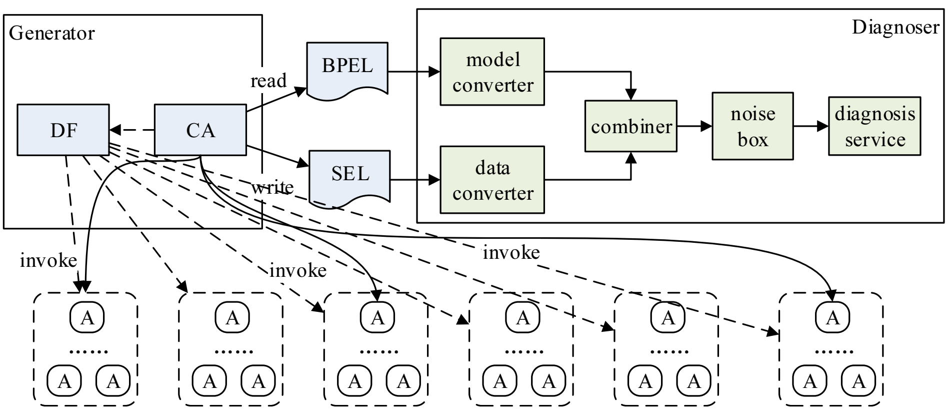

Fig. 2. Simulation experimental environment

4.2 Case Study

We illustrate application services with a running example, which implements a service composition application to plan a business travel for an employee. As shown in Fig. 3, the business travel process starts when a client requests this service process by application interface and it receives this request. Then it sends the employee information to the employee travel status service and requests a response. The employee travel status service returns the class standard (e.g. first class, business class and tourist class) of the employee to the service process according to the employee information. When the service process gets the information about the employee class standard and travel date, it sends the information to American Airlines service and Air China service. These two services retrieve ticket price by received information and return ticket price to the service process. Finally, the service process selects the cheaper ticket and sends the result to the service process. The service process informs the client agent of the cheaper ticket information by application interface. These invoked services are employed on the different computers. The composition service process uses corresponding virtual nodes on the virtualization layer to invoke them regardless of their physical locations and running environments.

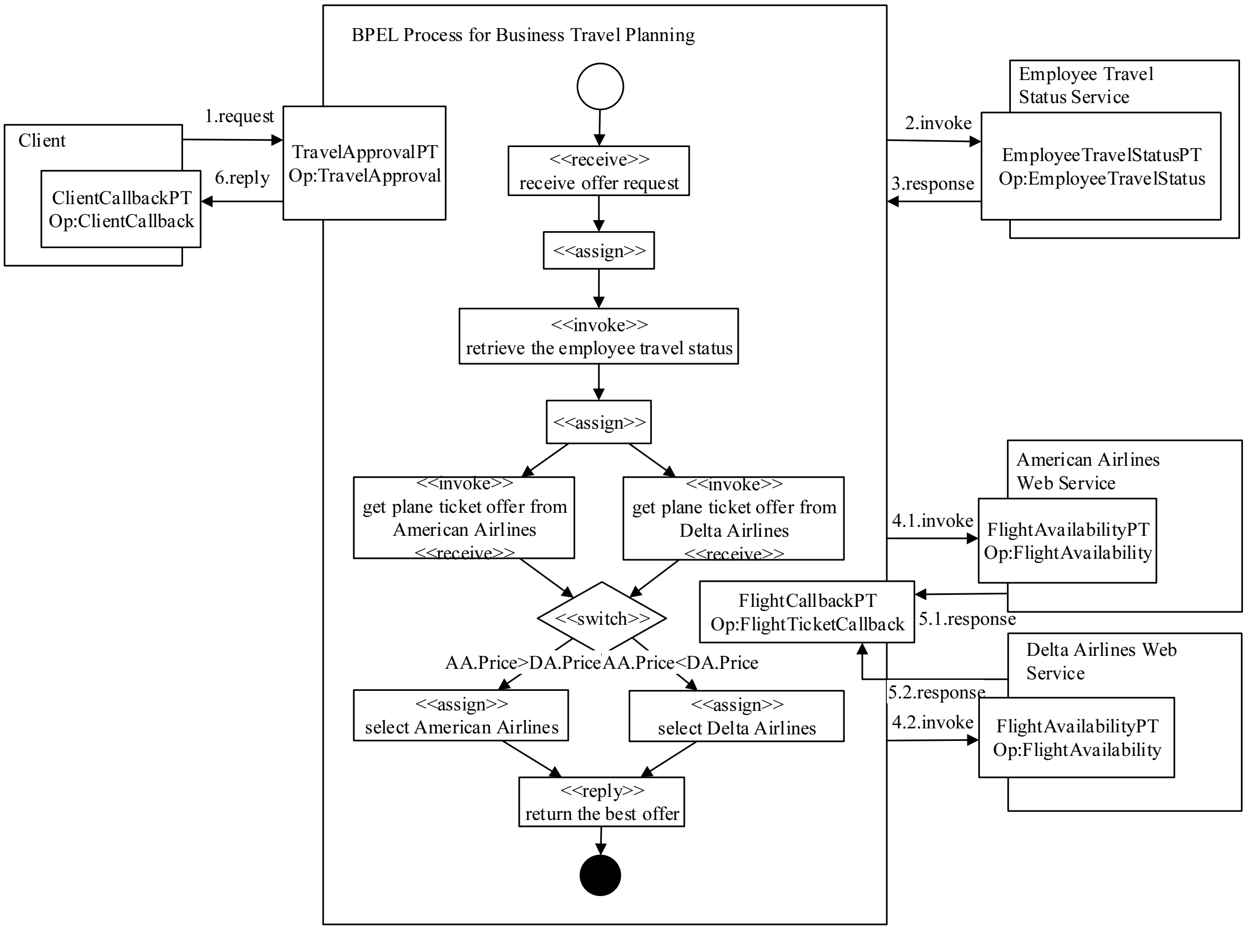

Fig. 3. BPEL process for business travel planning

This planning service actually implements in BPEL, which gives the formal specification for defining service process behaviors and service interaction protocols. BPEL is the de facto standard language for specifying and running actions within business process based on web service composition. It provides two kinds of activities for describing the internal behaviors: basic activity and structured activity. Structured activities prescribe the order in which a collection of activities take place. They describe how a business process is created by composing the basic activities [32].

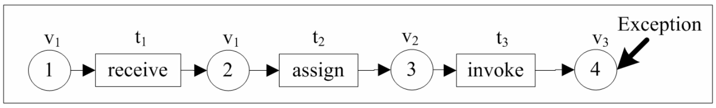

Fig. 4. Exception segment of business travel planning process

We first define the BPEL file of the *business travel planning service*. Suppose there is a semantic incompatibility between the composition service and the employee travel status service. The employee travel status service *ETS* returns client class standard information: (02, *vip*). '02' is the client ID. *vip* is the client service class. While the composition service receives it, and cannot find the class standard *vip*. Because the composition service uses the digit to represent the class standard. *vip* is converted into the digit and there is not the corresponding class. So, an exception occurs on the place 4 in Fig. 4 due to a semantic fault.

We apply our diagnosis model and algorithms to this business process. The case throws an exception, and observation sequence is $\sigma = \{(v_1, t_1, v_1), (v_1, t_2, v_2), (v_2, t_3, v_2), fail\}$. Here, t is an activity of business process. It is described as a behavior in our diagnosis model. And the input and output messages of these activities is denoted by the variables in our model.

Step 1. We use the **model converter** to convert BPEL file into the HMM-based model. Moreover, we apply the **data converter** to converting the running log into the HMM-based model. Finally, the **combiner** combines the two models into the diagnosis model. Here, given $\theta_m = 0.5$ and $\theta_r = 0.5$.

Step 2. We use the **noise box** to generate the final diagnosis model with 5% noise. Table 1 shows a part of behavior transition probability in the diagnosis model, Table 2 shows the emission probability and Table 3 shows the variable transition probability.

Table 1. Behavior transition probability

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
t_1	0	0.9960	0.0005	0.0005	0	0	0
t_2	0.0005	0	0.9570	0.0005	0.0010	0.0010	0.0005
t_3	0	0.0005	0	0.9960	0.0005	0.0010	0.0010
t_4	0	0.0005	0.0015	0	0.5830	0.4135	0
t_5	0	0	0	0.0010	0	0.1982	0.7451
t_6	0.0005	0.0005	0	0.0010	0.1947	0	0.0697
t_7	0	0.0417	0.0005	0.0010	0.0010	0.3856	0.0417

Table 2. Emission probability

	v_1	v_2	v_3	v_4	v_5	v_6	v_7
v_1	0.4895	0.5070	0.0005	0.0005	0.0005	0.0005	0.0005
v_2	0.0010	0.0010	0.9921	0.0010	0.0010	0.0010	0.0010
v_3	0.0556	0.0556	0.0556	0.0556	0.0556	0.0556	0.0556
v_4	0.0010	0.0010	0.0010	0.0010	0.9921	0.0010	0.0010
v_5	0.0005	0.0005	0.0005	0.0005	0.0005	0.4979	0.4986
v_6	0.0006	0.0006	0.0006	0.0006	0.0006	0.6383	0.0006
v_7	0.0007	0.0007	0.0007	0.0007	0.0007	0.0007	0.6824

Table 3. Variable transition probability

	v_1, v_1	v_1, v_2	v_2, v_3	v_4, v_5	v_5, v_6	v_5, v_7	v_6, v_6
t_1	0.9600	0.0005	0	0	0	0	0
t_2	0.0005	0.9560	0.0005	0.0005	0.0005	0	0.0010
t_3	0.0005	0	0.9970	0.0005	0.0005	0.0010	0.0000
t_4	0	0.0005	0.0005	0.9970	0	0.0010	0.0005
t_5	0.0005	0	0.0005	0	0.9955	0.0010	0.0005
t_6	0	0	0.0010	0.0010	0.0005	0.9965	0
t_7	0.0005	0	0	0.0005	0.0005	0	0.9553

Step 3. According to the diagnosis model, we use the **diagnosis service** to get a correct variable pair sequence ***VQ*** and a diagnosis solution of variables ***VS***. In here, $VA(v_1,v_1)=0.4895$, $VA(v_1,v_2)=0.5707$, $VA(v_2,v_2)=0.0010$, and $\Delta=0.05$. So, the fault occurs in the position t_3, $VS=\{(v_2,t_3,v_2)\}$. The probability of (v_2,v_3) is the maximum probability from v_2. So $VQ=\{(v_1,t_1,v_1),(v_1,t_2,v_2),(v_2,t_3,v_3)\}$. Finally, the diagnosis service gives an optimum behavior sequence $sq=\{t_1,t_2,t_3\}$ and a local diagnosis solution $LS=\{(v_2,t_3,v_2)\}$.

4.3 Evaluation

In our simulation experiment environment, we compare our diagnosis method with the Ardissono's method. The Ardissono's method uses the Petri nets to model the diagnosis models. Moreover, for evaluating the effectiveness of the methods, we generated six groups of BPEL files and SEL files. The noise proportions of generated diagnosis models are as follows: (1) 0%; (2) 1%; (3) 5%; (4) 10%; (5) 15% and (6) 20%.

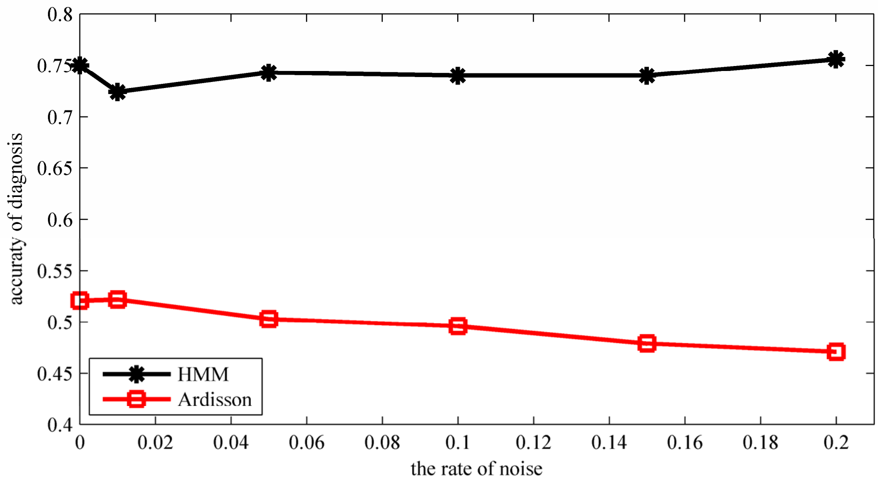

Fig. 5. Accuracy comparison between HMM-based method and Ardissono's method

Fig. 5 shows that the diagnosis accuracy of our method **HMM** is higher than the **Ardissono** without the rates of noise in the diagnosis models. In here, the diagnosis accuracy denotes the percentage of the number of correct diagnosis to the given six composition services. When the rate of noise is zero, the diagnosis accuracy of **HMM** is 23.8% more than **Ardissono**. When the rate of noise is 5%, the diagnosis accuracy of **HMM** is 19.6% more than **Ardissono**. When the rate of noise is 20%, the diagnosis accuracy of HMM is 27.5% more than **Ardissono**. The experimental result proof that **HMM** is more effective than **Ardissono** in the fault diagnosis. It is because our method considers the context of the service behaviors and incorporates the history data into diagnosis model.

In Fig. 6, we use three diagnosis data sets to build our diagnosis model: (1) **HMM** uses the BPEL file and service running log file to build the diagnosis model; (2) **model** only uses the BPEL file to build the diagnosis model; and (3) **reocrd** only uses the service running log file to build the diagnosis model. When the rate of noise is zero, the accuracy of **HMM** is same with **model** and **record**. When the rate of noise is 5%, the diagnosis accuracy of **HMM** is 1.6% more than **model**. When the rate of noise is 20%, the diagnosis accuracy of **HMM** is 12.7% more than **record** and 0.3% less than **model**. The experimental results shows that three data sets all achieve the good results. In addition, the diagnosis accuracy of **HMM** is more robust to various noise than **model** and **record** in most cases.

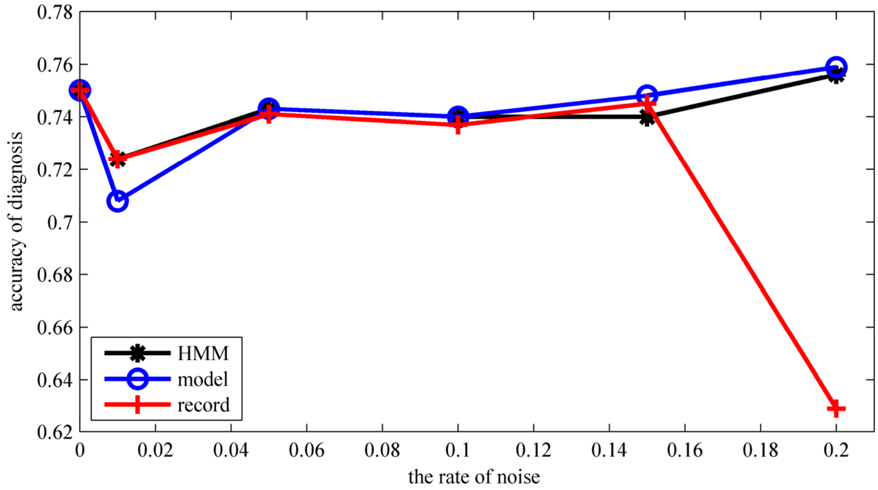

Fig. 6. Accuracy comparison of three diagnosis data sets

5 Related Work

In this section, we review two kinds of major methods about fault diagnosis of web service, including MBD methods and process history based methods.

5.1 MBD Methods

In order to enhance fault management in complex web services with the ability of reasoning on global failures of the overall service, Ardissono in [33] proposed a diagnostic framework for adding diagnostic capabilities to web services. The method adopts the grey-box models for individual web services, which are modeled in a component-oriented fashion. The diagnostic reasoning is performed partly by local diagnosis services and partly by a global diagnosis service. Unlike our diagnostic framework in this paper, the diagnostic system in [33] may have several global diagnosis services, where a lower level global diagnosis service is regarded as local diagnosis service by a higher diagnosis service. This approach could recursively partition web services into aggregations of sub-services, hide the details of the aggregation to higher-level services so that it could raise diagnostic efficiency and ensure the privacy of service. However, when most of the web services claim too coarsely, almost all web services could be faults [12].

Li et al. [16] proposed a diagnosis method for BPEL processes in a choreographed scenario, where a local diagnosis service translates the basic BPEL service and its local fault model into a CPN (colored Petri nets) model and a global diagnosis service includes the global fault model. According to three dependency relations proposed in [33], Li associates each dependency relation with a color propagation function to represent the data status. According to the color propagation functions, the diagnosis services recursively compute the diagnosis results until arriving at a final consistency. In [16], Li et al. assume the explicit fault models are existing. Unfortunately, detailed s fault models are not necessarily generally being satisfied in practice.

Yan et al. [12] modeled BPEL services as the synchronized automaton and diagnosed the service faults by reconstructing the execution trajectories based on the process model and the given observation. Yan's method assumes that faults are expressed by inconsistence between observation and process model [21]. In practice, this assumption does not always hold.

Benayas et al. [34] describes an autonomic Bayesian fault diagnosis architecture based on the big data analysis. Besides works mentioned above, some other researchers have applied MBD to the diagnosis of web service composition, like [21, 35-36] and so on.

5.2 Process History Based Methods

Meanwhile, many researchers used the approach of process history based methods to study the diagnostic problem [37]. For example, Kemper and Tepper [28] described a technique for finding faults by identifying and removing cycles from a simulation trace [28]. Dai et al. [26] proposed a method for diagnosing the faults of composite service based on the computation of the error propagation degree and fuzzy reasoning. Liu and Qiu [38] products an approach of service faults diagnosis based on conditional random fields.

This kind of method does not need an explicit service process model. It is easy to implement and able to handle noise effectively. However, its effectiveness is limited by the availability of only a finite sampling of historical process data. Moreover, most of these methods are hard to distinguish the fault from other unknown faults and provide an explanation for the faulty causes.

Table 4. Comparisons of diagnostic methods for web service

Methods	Advantages	Limitations	Existing	
consensus-based diagnosis	1. To distinguish the unknown faults; 2. to explain the propagation paths of faults.	A self-contained service mode.	[3, 10, 12, 17, 33, 36]	MBD Methods
Bayesian network-based diagnosis	1. A high diagnosis efficiency; 2. without a self-contained service model.	Manual intervention.	[25, 34]	
path analysis-based diagnosis	Without a self-contained service model.	1. To repeatedly execute the process in order to get the observed value; 2. pre-defined distinguish conditions.	[21]	
fault model-based diagnosis	To quickly find the faulty location by the fault model.	1. A self-contained fault model; 2. be unable to diagnose the undefined faults.	[16, 19, 39]	
probabilistic analysis-based diagnosis	1. Low time complexity; 2. high diagnosis efficiency.	1. Be unable to explain the propagation paths of faults; 2. the history data should have a high information coverage.	[6, 26, 38, 41]	Process History based Methods
path comparison-based diagnosis	1. To use the history execution trace to diagnose the service fault; 2. high diagnosis efficiency.	Be unable to diagnose the faults without coverage in the history data.	[28, 40]	

Table 4 shows a comparison of related works. MBD methods generally require a self-contained service model. Process history based methods are difficult to explain the failure causes. The proposed method in this paper incorporates historical process data into model-based diagnosis. It can overcome the limitations of the uncompleted process model and finite historical process data. Moreover, our method can get the asymptotically optimal diagnosis information with the increasing of history data.

6 Conclusion

In this paper, we present a new diagnosis architecture and a HMM-based diagnosis method for the fault diagnosis during the service process execution. Our diagnostic architecture decouples the diagnosis service components, so that it can share and reuse diagnosis service component to deal with the complex diagnosis processes. Hierarchical scalability is a feature of our method, which allows diagnosis system to integrate hierarchical diagnosis service components. It simplifies the diagnosis process, adds the diagnosis capability and guarantees the privacy of web services. The HMM-based method developed here constructs the diagnosis model based on the process model and the service running history data. The hybrid model-based diagnosis is another important feature of our method. It incorporates historical process data into model-based diagnosis for the web service, in order to overcome the limitations of the available priori knowledge. Based on these features, our method is more suitable to diagnose the faults of the web service composition including the less priori knowledge. Experimental results show that our method is effective to various noises in diagnosing the faults of service processes.

The proposed approach can be extended in the building of diagnosis model. In fact, an observation output occurrence trends to depend on not only the current state of the service but also the last state. A multiple-order HMM diagnosis model can improve the diagnosis accuracy. However, when we incorporate too many service states or activities into the diagnosis model, the diagnosis efficiency exponentially decreases. This introduces new challenges and requires a model that can cope with such difficult.

Acknowledgements

This paper is partially supported by the National Natural Science Foundation of China under Grant No. 61972053, No. 61603054, by the Scientific Research Foundation of Liaoning Education Department under Grant No. LQ2019016, No. LJ2019015, and by the Natural Science Foundation of Liaoning Province, China under Grant No. 2019-ZD-0505.

References

[1] W. Lijun, B. Xiaoying, Z. Lizhu, C. Yinong, A hierarchical reliability model of service-based software system, in: Proc. Computer Software and Applications Conference, 2009.

[2] G. Lamperti, M. Zanella, Context-sensitive diagnosis of discrete-event systems, in: Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011), 2011.

[3] L. Ardissono, L. Console, A. Goy, G. Petrone, C. Picardi, M. Segnan, Cooperative model-based diagnosis of web services, in: Proc.16th International Workshop on Principles of Diagnosis Pacific Grove, 2005.

[4] S. Yin, G. Wang, H.R. Karimi, Data-driven design of robust fault detection system for wind turbines, Mechatronics 24(2014) 298-306.

[5] S. Yin, G. Wang, X. Yang, Robust PLS approach for KPI related prediction and diagnosis against outliers and missing data, International Journal of Systems Science 45(2014) 1375-1382.

[6] X. Han, Z. Shi, W. Niu, K. Chen, X. Yang, Similarity-based Bayesian learning from semi-structured log files for fault diagnosis of web services, in: Proc. 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010.

[7] S.X. Ding, S. Yin, K. Peng, H. Hao, B. Shen, A novel scheme for key performance indicator prediction and diagnosis with application to an industrial hot strip mill, IEEE Transaction on Industrial Informatics 9(2013) 2239-2247.

[8] S.X. Ding, P. Zhang, S. Yin, E.L. Ding, An integrated design framework of fault-tolerant wireless networked control systems for industrial automatic control applications, IEEE Transactions on Industrial Informatics 9(2013) 462-471.

[9] L.M. Elshenawy, S. Yin, A.S. Naik, S.X. Ding, Efficient recursive principal component analysis algorithms for process monitoring, Industrial & Engineering Chemistry Research 49(2010) 252-259.

[10] L. Ardissono, R. Furnari, A. Goy, G. Petrone, M. Segnan, Fault tolerant web service orchestration by means of diagnosis, Software Architecture 4344(2006) 2-16.

[11] A. Liu, Q. Li, L.S. Huang, M.J. Xiao, FACTS: a framework for fault-tolerant composition of transactional web services, IEEE Transactions on Services Computing 3(2010) 46-59.

[12] Y. Yan, P. Dague, Y. Pencole, M.-O. Cordier, A model-based approach for diagnosing faults in web service processes, The International Journal of Web Services Research 6(2009) 87-110.

[13] K.-L. Peng, C.-Y. Huang, Reliability evaluation of service-oriented architecture systems considering fault-tolerance designs, Journal of Applied Mathematics 2014(2)(2014) 1-11.

[14] W. Gaaloul, S. Bhiri, M. Rouached, Event-based design and runtime verification of composite service transactional behavior, IEEE Transactions on Services Computing 3(2010) 32-45.

[15] O. Kopp, F. Leymann, D. Wutke, Fault handling in the web service stack, Service-Oriented Computing 6470(2010) 303-317.

[16] Y. Li, L. Ye, P. Dague, T. Melliti, A decentralized model-based diagnosis for BPEL services, in: Proc. 21st IEEE International Conference on Tools with Artificial Intelligence (ICTAI '09), 2009.

[17] M. Alodib, B. Bordbar, A model-based approach to fault diagnosis in service oriented architectures, in: Proc. 7th IEEE European Conference on Web Services, 2009.

[18] A. Carrera, C.A. Iglesias, J. Garcia-Algarra, D. Kolarik, A real-life application of multi-agent systems for fault diagnosis in the provision of an internet business service, Journal of Network and Computer Applications 37(2014) 146-154.

[19] Y. Li, T. Melliti, P. Dague, Modeling BPEL web services for diagnosis: towards self-healing web services, in: Proc. 3rd International Conference on Web Information Systems and Technologies (WEBIST'07), 2006.

[20] H.N. Lakshmi, H. Mohanty, Automata for web services fault monitoring and diagnosis, International Journal of communication and computer Technologies 3(2011) 13-18.

[21] W. Mayer, G. Friedrich, M. Stumptner, Diagnosis of service failures by trace analysis with partial knowledge, Service-Oriented Computing 6470(2010) 334-349.

[22] V. Venkatasubramanian, R. Rengaswamy, K. Yin, S.N. Kavuri, A review of process fault detection and diagnosis part III: process history based methods, Computers & Chemical Engineering 27(2003) 327-346.

[23] S. Yin, X. Li, H. Gao, O. Kaynak, Data-based techniques focused on modern industry: an overview, IEEE Transactions on Industrial Electronics 62(1)(2014) 657-667.

[24] S. Yin, S.X. Ding, X. Xie, H. Luo, A review on basic data-driven approaches for industrial process monitoring, IEEE Transactions on Industrial Electronics 61(2014) 6418-6428.

[25] H. Wang, G. Wang, A. Chen, C. Wang, C.K. Fung, S.A. Uczekaj, Modeling Bayesian networks for autonomous diagnosis of web services, in: Proc. Nineteenth International Florida Artificial Intelligence Research Society Conference, 2006.

[26] Y. Dai, L. Yang, B. Zhang, Z. Zhu, Exception diagnosis for composite service based on error propagation degree, in: Proc. 2011 IEEE International Conference on Services Computing (SCC 2011), 2011.

[27] S. Duan, H. Zhang, G. Jiang, X. Meng, Supporting system-wide similarity queries for networked system management, in: Proc. 2010 IEEE-IFIP Network Operations and Management Symposium, 2010.

[28] P. Kemper, C. Tepper, Automated trace analysis of discrete-event system models, IEEE Transactions on Software Engineering 35(2009) 195-208.

[29] K. Yang, G. OuYang, L. Ye, Research upon fault diagnosis expert system based on fuzzy neural network, in: Proc. WASE International Conference on Information Engineering, 2009.

[30] A.S. Naik, S. Yin, S.X. Ding, P. Zhang, Recursive identification algorithms to design fault detection systems, Journal of Process Control 20(2010) 957-965.

[31] S. Yin, X. Zhu, O. Kaynak, Improved PLS focused on key performance indictor related fault diagnosis, IEEE Transactions on Industrial Electronics 2014.

[32] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, S. Weerawarana, Business process execution language for web services, version 1.1, May, 2003

[33] L. Ardissono, L. Console, A. Goy, G. Petrone, C. Picardi, M. Segnan, enhancing web services with diagnostic capabilities, in: Proc. Third European Conference on Web Services (ECOWS'05), 2005.

[34] F. Benayas, A. Carrera, C.A. Iglesias, Towards an autonomic Bayesian fault diagnosis service for SDN environments based on a big data infrastructure, in: Proc. 2018 Fifth International Conference on Software Defined Systems (SDS), 2018.

[35] G. Lamperti, M. Zanella, EDEN: an intelligent software environment for diagnosis of discrete-event systems, Applied Intelligence 18(2003) 55-77.

[36] D.M. Muñoz, A. Correcher, E. García, F. Morant, Stochastic DES fault diagnosis with colored interpreted petri nets, Mathematical Problems in Engineering (2015) 1-13.

[37] S. Yin, X. Yang, H.R. Karimi, Data-driven adaptive observer for fault diagnosis, Mathematical Problems in Engineering 2012(2012). DOI: 10.1155/2012/832836.

[38] Y. Liu, L. Qiu, An approach to web service faults diagnosis based on conditional random fields, International Journal of Control and Automation 10(2)(2017) 129-136.

[39] G. Fan, A petri net-based byzantine fault diagnosis method for service composition, in: Proc. IEEE 36th Annual Computer Software and Applications Conference (COMPSAC), 2012.

[40] G.K. Mostefaoui, On modeling and developing self-healing web services using aspects, in: Proc. 2nd International Conference on Communication System Software and Middleware and Workshops (COMSWARE 2007), 2007.

[41] X. Han, B. Li, K.F. Wong, Z. Shi, Exploiting structural similarity of log files in fault diagnosis for web service composition, CAAI Transactions on Intelligence Technology 1(2016) 61-71.