Indian Journal of Science and Technology, *Vol 9(25), DOI: 10.17485/ijst/2016/v9i25/85880, July 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Managing Dependencies for a Hierarchical Service-based System

Sridevi Saralaya[1*], Rio D'Souza[1] and Vishwas Saralaya[2]

[1]Department of Computer Science, St. Joseph Engineering College, Visvesvaraya Technological University, Vamanjoor, Mangalore – 575028, Karnataka, India; sridevisaralaya@gmail.com, rio@ieee.org
[2]Department of Microbiolgy, Kasturba Medical College, Manipal University, Mangalore - 575001, Karnataka, India; vishwassaralayak@gmail.com

Abstract

This study proposes a hierarchical model of a Service-Based System (SBS) to represent horizontal and vertical dependencies within a SBS. In order to detect root-causes and conduct impact-analysis of anomalies occurring in a SBS, we represent the SBS as a multi-layer system consisting of Business Process Management (BPM) layer, Service Composition and Coordination (SCC) layer and Service Infrastructure (SI) layer using Hypergraphs. The intra-layer and inter-layer relationships are depicted by hyperedges that effectively depicts n-ary relationships which are not possible with simple graphs. Using hypergraphs and hyperedges we have effectively represented intra-layer horizontal time dependencies and inter-layer vertical time and resource dependencies. Horizontal time dependencies help us to analyse the impact of a time delay on related entities of the same layer. Vertical resource dependencies help in root-cause analysis of the time delay. Vertical time dependencies aid in finding the impact of service delay on activities of the business layer. Our approach based on hypergraph helps to model relationships of a SBS from multiple perspectives using hyperedges. The proposed approach meticulously represents various vertical and horizontal dependencies between elements of a SBS and can be effectively utilised to identify root-cause of an anomaly and its impact on related entities of a hierarchical SBS.

Keywords: Dependency, Hierarchical Service-Based System, Impact-Analysis, Multi-Level Hypergraphs, Root-Cause Analysis

1. Introduction

Business processes are modelled and realised by a sequence of activities in a workflow which contribute to attain the objectives of the business process. The execution of each activity requires a single service or a set of services. Services implementing activities require resources for their execution. A resource can be described as any entity required by a service for its execution such as human resources, IT resources, IS resources, automated devices, printers/plotters, a database, a document etc[1]. In a composite business process implemented as a Service-Based System (SBS), dependencies exist between the business process, component services and the resources required for their execution which can be considered as a multi-tiered system. We visualise the SBS as a multi-tier system consisting of Business Process Management (BPM) layer, Service Composition and Coordination (SCC) layer and Service Infrastructure (SI) layer[2]. A dependency is a relationship between entities, where an entity can be an activity in the business process workflow, a service implementing an activity or a resource required for execution of a service. Often dependency information is not explicitly available and is implicit in the Service Level Agreements (SLAs), knowledge of domain experts, service descriptions, workflows and the composite business process[3]. This implicit knowledge has to be explicitly represented in a dependency model for various reasons such as failure handling, root-cause analysis, impact analysis, SLA violation analysis, handling service evolutions etc. A major building block for root-cause and impact analysis is the creation of the dependency model.

**Author for correspondence*

Dependencies exist between entities within a layer in the same hierarchical level and between entities across the layers in different hierarchical levels. The dependency model has to explicitly capture the relationships in and across the layered elements of a SBS. Existing studies concentrate on either inter-layer or intra-layer relationships between elements in the context of SLA violation. Such approaches lack the capability to represent relationships from multiple facets. We review some of the works which represent dependencies using dependency graphs or dependency matrix. A Study by Sensarma et al. discusses the advantages of using graphs for detecting anomalies[4].

Winkler and Schill discuss the SLA violations that arise due to dependencies between atomic services in a service composition[3]. The types of dependencies considered are timing and resource (data) dependency. The authors term the dependencies between atomic services as horizontal dependencies and dependency between an atomic service and the composite service as vertical dependency. A formal model to depict the dependencies between services is created based on information contained in the SLAs. The focus is on analysing dependencies to assist in identifying SLA violations and not for root-cause or impact analysis.

A model for formalizing dependencies in SLAs is proposed by Bondestaff et al.[5] Dependencies are considered for evaluating SLA violations to the composite business process from the atomic services. Time and cost are the categories considered for analysing vertical dependencies. The study considers only the vertical dependencies and not horizontal dependencies. Another study discusses data and control dependencies in a business process based on control structures of the workflow[6]. The Dependency model is derived based on parameter, pre-condition and effects between semantically annotated business activities. The study considers dependencies for runtime handling of sequencing constraints between atomic services i.e. only considers horizontal dependencies and not vertical dependencies. OWL-DL and Meta-model based modelling approaches have been compared by Sell et al. analyse the dependencies from the view-point of re-negotiation of SLAs when problems occur during the execution of a business process[7]. Types of dependencies considered are After, Before, Parallel, Requires and Alternative. The study considers dependencies from the point of SLAs and not for analysis when anomalies are observed. Keller et al. consider dependencies between applications and their hosting servers in a distributed system[8].The authors organise dependencies in distributed systems from lower levels to higher levels as intra-package, intra-system, inter-system, intra-domain and inter-domain. An impact-analysis model was proposed to identify the region of a system that may be affected due to service evolutions[9].The authors propose service dependency graph and relationship matrix to analyse the relationship between services. The dependencies considered between services are based on input required and output produced. The authors do not consider temporal or infrastructure dependencies. The study by Omer et al. automatically extracts dependencies from abstract service descriptions which assist in constructing the model of a business process[10]. The authors consider horizontal dependencies for automatically building composite web service but not for root-cause or impact analysis. A series of studies conducted develop dependency model based on hypergraphs to verify correctness of dependencies during construction of multi-tenant applications on the cloud[11-14]. The dependencies are considered on three hierarchical service levels which are business-independent level, business-dependent level and composite business level. An incidence matrix is used to represent information contained in the hypergraph. The dependencies between service levels are realized by reduction technique. The studies consider dependencies for creation of multi-tenant applications on the cloud and not for analysing causes and their impact when faults occur.

It is obvious that a dependency graph or dependency matrix is only suitable for expressing dependencies with limited information. They do not provide enough expressivity for dependency models for the management of multiple dependencies in a layered SBS. There is a substantial gap in representing the SBS in the form of hierarchical layers and the intra and inter-layer dependencies between its layers to assist in root-cause and impact analysis. Limitations of the graph and matrix in representing only binary relations and their inability to denote the n-ary relation can be overcome by Hypergraphs.

In our study a dependency model is developed to assist in root-cause and impact analysis considering both horizontal and vertical relationships. Our main contributions for representing dependency information and their analysis is – development of a model to represent the hierarchical layered SBS that helps to: 1. Identify elements in each layer and 2. To depict horizontal and vertical dependencies in and across layers of the SBS.

The paper is organised as follows: Section 2 describes the types of dependencies that exists in a hierarchical SBS, in Section 3 we explain the proposed hierarchical dependency model for SBS, Section 4 explains the usage of the dependency model for root-cause and impact-analysis. In Section 5 we discuss the creation of the dependency model for a Blood Testing Centre and we conclude in Section 6.

2. Dependency Types

A dependency can be described as a relation between entities where a modification or variation in one entity may affect other entities. The goal of creating a dependency model is to describe the elements that make up the layers of an SBS and the dependencies between them. Service dependencies are the cause for the fact that events regarding single services influence other services in a service composition. The nature of the dependencies is analysed based on their occurrences within the SBS layers. We can classify the dependencies as (Figure 1) :

- Horizontal dependencies (intra-layer).
- Vertical dependencies (inter-layer).

2.1 Horizontal Dependencies

The dependencies which occur within elements of the same layer of a hierarchical layered SBS are denoted as horizontal dependencies also termed as intra-layer dependencies. Horizontal dependencies can be Intra-activity dependency (BPM layer), Intra-Service dependency (SCC Layer) or Intra-Resource dependency (SI Layer). Temporal Intra-Layer dependency (e.g. Intra-Activity or Intra-Service) specifies the timing relationship between entities of a layer. Horizontal dependencies can be further classified as: 1. Intra-Service Temporal dependency - between a preceding service S_i and a succeeding service S_{i+1} specifies that S_{i+1} can start executing only after S_i has completed execution. 2. Intra-Layer Data dependency - between two services S_i and S_{i+1} indicates that the output data produced by service S_i acts as input to service S_{i+1}.

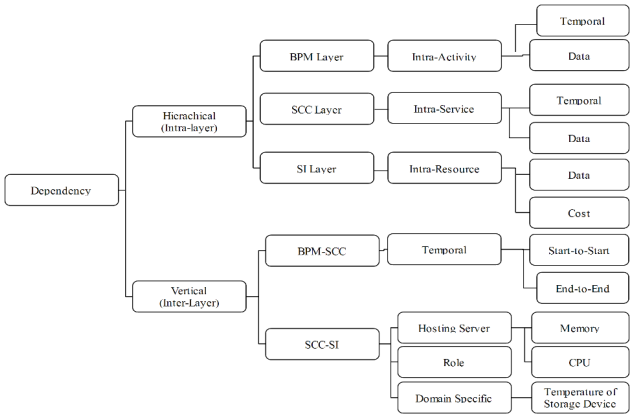

Figure 1. Taxonomy of dependencies in a SBS.

2.2 Vertical Dependencies

The dependencies which occur between elements across layers of the SBS are denoted as vertical dependencies also termed as inter-layer dependencies. Vertical dependencies for a hierarchical SBS can be further classified as: 1. BPM_SCC dependency - between the Business Process and Service Composition layer. The execution of a business process can start only when the first service in the SCC layer starts executing, likewise the execution of the business process is complete only when the last service in the composition completes execution. 2. SCC_SI dependency – between the Service Composition and Service Infrastructure layer. Every service needs some resource for its execution. For example web services are hosted on a computing node. The parameters of the hosting node such as CPU load and available memory affect the services hosted on the node. Application specific infrastructure such as refrigerator for storage of medicines or pages/ink of a printer can also be considered as parameters of the infrastructure layer.

3. Proposed Hierarchical Dependency Model for SBS

The hierarchical dependency model for SBS has to be created at design-time based on the structural workflow pattern of the business process. The requirements to be fulfilled by this model are: 1. It has to represent horizontal dependencies between entities of a layer; and 2. Should represent vertical dependency across layers.

Edges of a simple graph can connect only two vertices, whereas edges of a hypergraph known as hyperedges can contain a set of vertices. Our approach represents relationship information within a layer and between layers of a hierarchical SBS by a dependency model based on the concept of hypergraphs[15]. Hypergraph, an extension of graph theory, is a discrete mathematical structure. Hypergraphs can be used to represent multi-function

and multi-level relationships[16]. Hypergraphs are based on Granular Computing paradigm which helps to view the same problem from different perspectives[17].

A granule is a set of elements which share some similar properties. A set of entities which have some relation and can be considered as a group are regarded as a granule. In our study we consider the elements of a layer such as activities in the top layer, services in the middle layer and resources of the bottom layer as granules. Granules in each layer, even though relatively independent, are related to each other by some common characteristic. The relationships between entities inside a granule represent the intra-level or horizontal dependency.

3.1 Basics of Hypergraph

A hypergraph given as H = (V, E) is defined by a set of vertices V and a set of hyperedges E among the vertices. Each edge e ∈ E is a subset of vertices and every vertex v ∈ V. Pins represents the vertices in an edge E and is denoted by pins [E]. The size of an edge is the number of its pins i.e. S_e = |pins [E]|. The set of edges connected to a vertex v is denoted by edges [v]. The degree of a vertex is the number of edges it is connected to i.e. d_v= |edges[v]|[15].

3.1.1 Entity Space and Relation

An entity space is a system (O, RL), where:

- O is finite nonempty set of entities which may be a set of activities A = {a_1, a_2, a_3,.., a_n} of the business layers or a set of services S= {s_1, s_2, s_3, .., s_n} of the service composition layer or a set of resources R = {r_1, r_2, r_3, .. , r_n} of the infrastructure layer.
- RL is a finite nonempty set of relations RL = {rl_1, rl_2, rl_3, .. , rl_n } among the entities in O. For each rl_i ∈ RL, rl_i ⊆ O × O × … × O where i ≤ n. For (s_1, s_2, s_3, .., s_n) ⊆ O, if (s_1, s_2, s_3, .., s_n) ⊆ rl_i, then there is an i-ary relation r_i on (s_1, s_2, s_3, .., s_n) as shown if Figure 2.

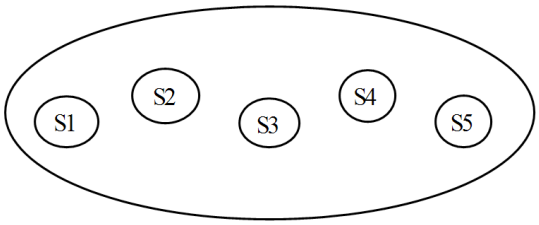

Figure 2. An example of a single-layer SBS.

In the entity space shown in Figure 2, the set of entities which have a relation rl ∈ RL can be considered as a unit and forms a granule. The smallest granule is a single entity and the largest is the set of all entities in the entity space. A vertex of the hyergraph corresponds to an entity and a hyperedge corresponds to the entities which are related through rl_i, highlighted portion to be removed. comma to be replaced with full stop. When all related vertices are combined together to form a hyperedge, a single-layer model from one view-point or perspective is formed.

3.2 Single-Layer Model Construction

A layer of a SBS is a granule of a hypergraph consisting of related entities. A granule belongs to a single-layer and is an assimilation of similar entities which help to resolve complications from one perspective. Hence in-order to create a hierarchical model of SBS we need three layers one each for BPM, SCC and SI layer. The BPM layer consists of entities which are activities of the workflow from the business perspective. The SCC layer consists of related entities which are services implementing activities of the BPM layer. The SI layer will consist of resources which are entities required for service execution.

Directed edges can be used to represent relationships between entities in a level known as intra-layer or horizontal dependencies as shown in Figure 3. A hyperarc or directed hyperedge is an ordered pair, E = (P, Q), of (possibly empty) disjoint subsets of vertices; P is known as the tail of E while Q is known as its head. The tail and the head of hyperarc E are denoted as T (E) and H (E), respectively. The directed edge e_2 (S_2, S_3) connecting S_2 and S_3 represents horizontal time dependency between the two services. The tail of the edge e_2 is represented with a value of -1 and its head with a value of 1 in the relation matrix. The horizontal time dependency between S_2 and S_3 indicates that S_3 can start only after S_2 has completed its execution. The relation matrix of a single-level hypergraph H shown in Table 1 is a n*n matrix a_{ij} defined as follows:

$$aij = \begin{cases} -1 \; if \; vi \in T(Ej) \\ 1 \; if \; vi \in H(Ej) \\ 0 \; otherwise \end{cases}$$

3.3 Two-layer Model Construction

A hypergraph which has directed edges is known as a directed hypergraph. A Backward hyperarch also known as B-arc, is a hyperarch E = (T (E), H (E)) with |H (E)| = 1. A Forward hyperarc, also known as F-arc, is a hyperacrc

E = (T (E), H (E)) with |T (E)| = 1, illustrated in Figure 4 (a) and (b) respectively[18].

B-arc and F-arc can be used to represent relationships between entities across two levels of a hierarchical SBS, known as inter-layer or vertical dependencies as shown in Figure 5. The tail of F-arc e_1 originates at service S_1 in the SCC layer and its head connects resources r_1 and r_2 in the SI layer, represents vertical resource dependency between service S_1 and resources r_1 and r_2. In order to differentiate between horizontal and vertical dependencies, we represent the tail of an edge representing vertical dependency with a value of -2 and its head with a value of 2. The tail of F-arc e_1 is represented with a value of -2 and its head with a value of 2. The F-arc e_1 indicates vertical resource dependency between service s1 and resources r_1 and r_2 i.e. the execution of service S_1 depends on the parameters of resources r_1 and r_2 such as CPU load, memory availability, temperature etc. The B-arc e_3 has S_3 and S_4 as its tail and r_4 as its head, indicates that service S_3 and S_4 require resource r_4 for their execution. The paucity of simple graphs to illuminate n-ary relationship is overcome by F-arc and B-arc

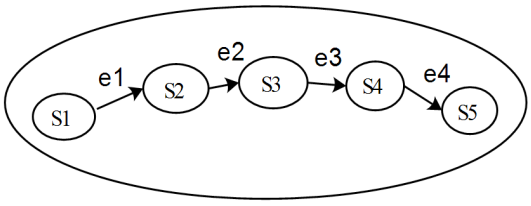

Figure 3. Relationship between entities in a single-level SBS.

Table 1. Relation Matrix for a single-level model

	s_1	s_2	s_3	s_4	s_5
s_1	0	1	0	0	0
s_2	-1	0	1	0	0
s_3	0	-1	0	1	0
s_4	0	0	-1	0	1
s_5	0	0	0	-1	0

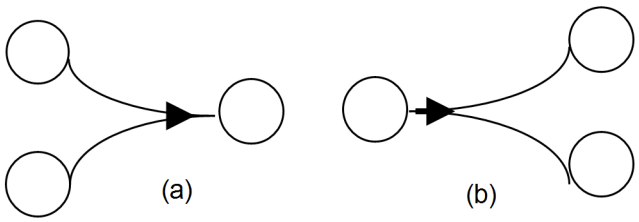

Figure 4. **(a)** B-arc **(b)** F-arc.

as represented by edges e_1 and e_3 respectively. The relation matrix of a two-layer SBS shown in Table 2 is an n*n matrix a_{ij} defined as follows:

$$aij = \begin{cases} -2 \text{ if } vi \in T(Ej) \\ 2 \text{ if } vi \in H(Ej) \\ 0 \text{ otherwise} \end{cases}$$

3.4 Multi-layer Model Construction

A Hierarchical Dependency-aware Model for SBS can be constructed by first creating single-level models and then assimilating them into a hierarchical model by mapping relationship between entities across layers using hyperedges. A set of hyperedges can be created to depict the connection between entities of a hierarchical layer of an SBS. The entire set of granules in each layer and their relationships can be viewed as a comprehensive representation of the hierarchical SBS.

B-arc and F-arc can be used to elucidate inter-layer vertical dependencies between BPM and SCC layer and

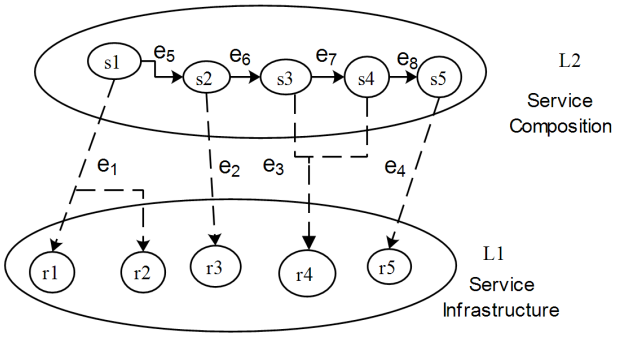

Figure 5. A two-layer model of the SBS.

Table 2. Relation matrix for a two-level model of SBS

	s_1	s_2	s_3	s_4	s_5	r_1	r_2	r_3	r_4	r_5
s_1	0	1	0	0	0	2	2	0	0	0
s_2	-1	0	1	0	0	0	0	2	0	0
s_3	0	-1	0	1	0	0	0	0	2	0
s_4	0	0	-1	0	1	0	0	0	2	0
s_5	0	0	0	-1	0	0	0	0	0	2
r_1	-2	0	0	0	0	0	0	0	0	0
r_2	-2	0	0	0	0	0	0	0	0	0
r_3	0	-2	0	0	0	0	0	0	0	0
r_4	0	0	-2	-2	0	0	0	0	0	0
r_5	0	0	0	0	-2	0	0	0	0	0

between SCC and SI layer. As explained in section 3.3, if the values of the relation matrix to represent vertical dependencies is 2 (for H (E_i)) and -2 (for T (E_i)), then we will not be able to differentiate the inter-layer relation between BPM and SCC layer and SCC and SI layer. To overcome this impediment, we represent the T (E_i) with the value -2 and its H (E_i) with dep_type which can take the value of V_T or V_R signifying vertical time and vertical resource dependency respectively. Likewise, dep_type will have the value H_T to represent horizontal time dependency as given below.

$$aij = \begin{cases} -1 \; if \; vi \in T(Ej) \; intra-layer \\ -2 \; if \; vi \in T(Ej) \; inter-layer \\ dep_{type} if \; vi \in H(Ei) \\ 0 \; otherwise \end{cases}$$

Figure 6 illustrates the hypergraph representing the model of hierarchical SBS. The tail of F-arc e_9 originates at activity a1 in the BPM layer and its head connects S_1 and S_2 in the SCC layer representing vertical time dependency between activity a1 and services S_1 and S_2. The tail of F-arc e_9 is represented with a value of -2 and its head with a value of V_T. The F-arc e_9 indicating vertical time dependency between activity a_1 and services S_1 and S_2 represents a start-to-start dependency between a_1 and S_1 and end-to-end dependency between a_1 and S_2. The F-arc e_1 originating at service S_1 in the SCC layer depicts vertical resource dependency between service S_1 and resources r_1 and r_2 in the SI layer. The tail of F-arc e_1 is represented with a value of -2 and its head with a value of V_R.

Consequently, information contained in the hypergraph is articulated into a Relation matrix shown in Table 3, which depicts the relationship between entities and layers of a SBS. As evident, there is a one-to-one correspondence between the hypergraph and the relation matrix.

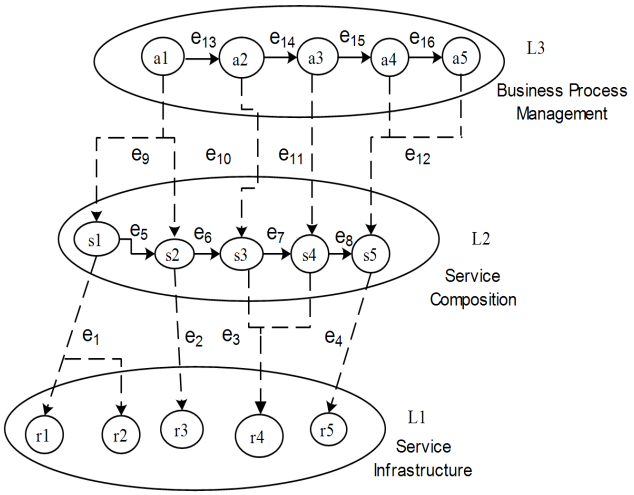

Figure 6. A hierarchical dependency model of the SBS.

Table 3. Relation matrix for the hierarchical SBS

	a_1	a_2	a_3	a_4	a_5	s_1	s_2	s_3	s_4	s_5	r_1	r_2	r_3	r_4	r_5
a_1	0	H_T	0	0	0	V_T	V_T	0	0	0	0	0	0	0	0
a_2	-1	0	H_T	0	0	0	0	V_T	0	0	0	0	0	0	0
a_3	0	-1	0	H_T	0	0	0	0	V_T	0	0	0	0	0	0
a_4	0	0	-1	0	H_T	0	0	0	0	V_T	0	0	0	0	0
a_5	0	0	0	-1	0	0	0	0	0	V_T	0	0	0	0	0
s_1	-2	0	0	0	0	0	H_T	0	0	0	V_R	V_R	0	0	0
s_2	-2	0	0	0	0	-1	0	H_T	0	0	0	0	V_R	0	0
s_3	0	-2	0	0	0	0	-1	0	H_T	0	0	0	0	V_R	0
s_4	0	0	-2	0	0	0	0	-1	0	H_T	0	0	0	V_R	0
s_5	0	0	0	-2	-2	0	0	0	-1	0	0	0	0	0	V_R
r_1	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0
r_2	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0
r_3	0	0	0	0	0	0	-2	0	0	0	0	0	0	0	0
r_4	0	0	0	0	0	0	0	-2	-2	0	0	0	0	0	0
r_5	0	0	0	0	0	0	0	0	0	-2	0	0	0	0	0

4. Application of Hierarchical SBS Dependency Model

The intention of developing the hierarchical dependency model for the SBS is to assist in root-cause and impact-analysis when anomalous behaviour is observed by monitoring the execution of the SBS. When anomalies are observed the services in the SCC layer have to be replaced with services offering similar functionality[19].In the following sections we elucidate the root-cause and impact-analysis algorithms.

4.1 Root-Cause Analysis

It helps to identify the actual cause of an anomaly in a SBS. If service S_i does not complete its execution at the required end_time as specified in the SLA, an anomaly is observed and the root-cause of this anomaly has to be identified based on its inter-layer and intra-layer dependencies as given by Algorithm 1. The inputs to the algorithm are the anomalous service and the Relation matrix. The output provided by the algorithm is a list containing the entities which could be the cause for deviation of service S_i. Line 4 to 9 in the algorithm identifies the inter-layer dependencies. Line 4 extracts the entities which have vertical resource dependencies with S_i. The status of these resources (such as CPU load and available memory) which are logged are extracted to detect if they had surpassed threshold limits. Such resources are added to the Root_Cause_List. Line 10 to 14 locates the intra-layer dependencies. The end_time of the antecedent service of S_i is verified to check if it was delayed. If yes, it is added to the Root_Cause_List.

Algorithm 1. To identify Root-Causes based on information from Relation Matrix

1. Input: The anomalous service S_1, Relation matrix
2. Output: Root_Cause_List
3. //Analyse Inter-layer (Vertical resource) dependency for root cause
4. Resource_Set = in Row S_1, identify the column which have the value V_R
5. Extract the status of the Resource_Set from the log file between the start_time and end_time of the anomalous service S_i.
6. For all resources in the Resource_Set
7. If Status of the Resource has exceeded threshold limits, add it to Root_Cause_List
8. End If
9. EndFor
10. //Analyse Intra-layer (Horizontal time) dependency for root cause
11. Antecedent_Service = in Row S_i, identify the column which have the value -1
12. In the log check whether observed end_time of Antecedent_Service is equal to SLA end_time
13. If not then delay in end_time of antecedent may also be a probable cause for delay of anomalous service
    Add Antecedent_Service to Root_Cause_List
14. End if

4.2 Impact Analysis

It helps to identify the impact or effect of an anomaly on related entities in a SBS. If service S_1 does not complete its execution at the required end_time, an anomaly is observed and the impact of such deviation on intra and inter-layer related entities has to be identified. Service S_{1+1} is directly dependent on S_1, services S_{1+2} to S_n are indirectly dependent on S_1 i.e. when there is a delay in the end_time of service S_1, it effects the start_time of service S_{1+1}. Hence S_{1+1} is directly impacted due to delay in S_1. Similarly services S_{i+2} to S_n will not be able to start executing at the stipulated start_time. Such dependencies can be obtained from the relation matrix as given by algorithm 2. Intra-layer time dependent entities are obtained from line 3 to 11. Delay in the end_time of a service (SCC layer) also has vertical time impact on the activities (of BPM layer) the service is a part of. Inter-layer impacted activities are obtained from line 12 to 20.

Algorithm 2. To identify Impact Analysis based on information from Relation Matrix

1. Input: the anomalous service S_1, relation matrix
2. Output: the list of impacted entities Impact_List
3. //Analyse intra-layer impact (horizontal time dependency)
4. Direct_Dependent_Entity = in row S_1, identify the column which has the value H_T
5. Impact_List = Direct_Dependent_Entity
6. //For Intra-layer indirect dependencies (horizontal time dependency)
7. For i= Direct_Dependent_Entity to total_no_of_Services
8. Indirect_Dependent_Entity = in the row of Direct_Dependent_Entity identify the column which has the value H_T
9. Add Indirect_Dependent_Entity to Impact_List

10. Direct_Dependent_Entity = Indirect_Dependent_Entity
11. EndFor
12. //Analyse inter-layer impact (vertical time dependency)
13. Direct_Dependent_Entity = In row S_1, identify the column which has the value -2
14. Add Direct_Dependent_Entity to Impact_List
15. //For Inter-layer indirect dependencies (horizontal time dependency)
16. For i= Direct_Dependent_Entity to total_no_of_activities
17. Indirect_Dependent_Entity = In the row of Direct_Dependent_Entity identify the column which has the value H_T
18. Add Indirect_Dependent_Entity to Impact_List
19. Direct_Dependent_Entity = Indirect_Dependent_Entity
20. Endfor

5. Experiment and Analysis

We consider a Blood Testing Centre as a business process implemented as a SBS. We visualize the elements of the Blood Testing Centre as a hierarchical SBS. The granules Sample Collection and Preservation, Testing and Validation and Report and Billing which have common characteristics are abstract activities of the business process which form the BPM layer is shown in Figure 7. The service granules responsible for implementing the activities of the top layer are integrated to form the SCC layer. The granules of the infrastructure layer are the computer nodes, automated devices (such as ELISA Processor, ELISA Reader), storage devices and humans responsible for invoking services.

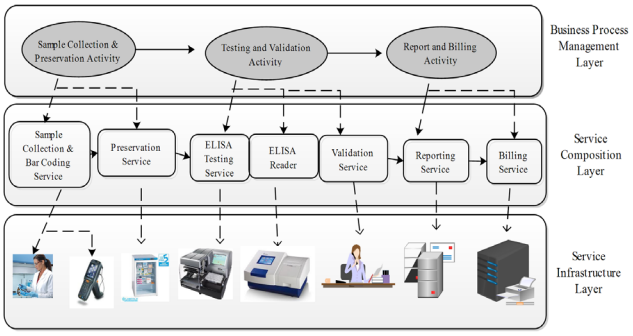

Figure 7. A Blood Diagnostic Centre visualised as a hierarchical SBS.

Each activity of the Blood testing centre requires one or more services which are considered as entities of the SCC layer. The first activity 'Sample Collection and Barcoding' requires two services for its execution – Sample Collection and Bar-coding Service and Preservation Service. There exists vertical temporal dependency (V_T) of type Start-to-Start between the first activity and Sample Collection Service, likewise there exists end-to-end temporal dependency between the first activity and Preservation service. Every service in the middle level requires infrastructure for its execution. For example, the ELISA Testing service, ELISA Reader service and Validation service (augmented together contribute to the second activity Testing and Validation of the BPM layer) require infrastructure such as automated ELISA processor, ELISA Reader and the human role 'Consultant Microbiologist' for certification of the results. Vertical Resource dependency (V_R) exists between the services in the SCC layer and resources of the SI layer.

Horizontal Time dependency (H_T) exits between the three activities in the BPM layer and between services in the SCC layer. For example the output produced by the ELISA processor is input to the ELISA Reader which plots an Optical Density (OD) graph which is used to read off OD values for blood test samples. These values are analysed by the role Consultant Microbiologist to certify the result of the blood test. ELISA Reader cannot be executed until ELISA processor completes execution, similarly Consultant Microbiologist cannot certify the result until the ELISA Reader plots the graph. This demonstrates an example of intra-layer horizontal time dependencies in the SCC layer. Table 5 gives the relation matrix of the SBS. Abbreviations for entities used in Table 5 are given in Table 4.

The Blood Test Centre can be realised as a Service-Based application (SBS) using BPEL 2.0 engines such as Activiti or Cammunda (https://camunda.com/). During execution of the SBS the timing of each service can be obtained using Aspect Oriented Programming (AOP) as explained in our earlier study[20] or by monitoring the timing of start and end events of each service based on event processing technology[21]. Suppose the ELISA Reader service (S_4) does not complete as per the end_time specified in the SLA. In such a case the root-cause for its deviation and impact of the delay has to be identified on the related entities of the SBS. Algorithm 1 is invoked to identify the root-causes. The inter-layer reasons are analysed by identifying the columns which have the value V_R in the row

Table 4. Abbreviations used in Table 5

Entity	Abbreviations
Sample Collection and Preservation Activity	a_1
Testing and Validation Activity	a_2
Report and Billing Activity	a_3
Sample Collection and Bar Coding	s_1
Preservation Service	s_2
ELISA Testing Service	s_3
ELISA Reader Service	s_4
Validation Service	s_5
Reporting Service	s_6
Billing Service	s_7
Sample Collection	r_1
BarCoder	r_2
Specimen Storage Refrigerator	r_3
ELISA Processor	r_4
ELISA Reader	r_5
Consultant Microbiologist	r_6
Reporting Server	r_7
Billing Server	r_8

S_4 (which is r_5), the parameters of the resource r_5 such as memory are analysed in the log to verify if it has exceeded the threshold limits. If threshold limits are exceeded the resource parameter is considered as one of the causes for delay in completion of service S_4. The intra-layer causes are identified by checking whether the antecedent service of the anomalous service S_4 had violated its end_time requirement specified in SLA. The antecedent service is identified by extracting the column which has the value -1 in row S_4, which is service S_3. Hence the log is verified to check whether service S_3 had exceeded its required end_time. If yes, delay in S_3 will also be considered as a cause for deviation of service S_4.

Algorithm 2 is invoked to identify the intra and inter-layer time impact analysis. In order to analyse intra-layer direct time impact, extract the column which has the value H_T in row S_4 (which is S_5) and hence service S_5 will be directly affected if there is a delay in S_4. S_5 is added to the Impact_List. To obtain the entities which are indirectly affected, in the row of S_5 obtain the column which have the value H_T (which gives service S_6), likewise service S_7 is also added to the Impact_List. In order to analyse inter-layer direct time impact, extract the column which has the value -2 in row S_4 (which is activity a_2). Activity

Table 5. Relation matrix for the Blood Diagnostic Hierarchical SBS

	a_1	a_2	a_3	s_1	s_2	s_3	s_4	s_5	s_6	s_7	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8
a_1	0	H_T	0	V_T	V_T	0	0	0	0	0	0	0	0	0	0	0	0	0
a_2	-1	0	H_T	0	0	V_T	V_T	V_T	0	0	0	0	0	0	0	0	0	0
a_3	0	-1	0	0	0	0	0	0	V_T	V_T	0	0	0	0	0	0	0	0
s_1	-2	0	0	0	H_T	0	0	0	0	0	V_R	V_R	0	0	0	0	0	0
s_2	-2	0	0	-1	0	H_T	0	0	0	0	0	0	V_R	0	0	0	0	0
s_3	0	-2	0	0	-1	0	H_T	0	0	0	0	0	0	V_R	0	0	0	0
s_4	0	-2	0	0	0	-1	0	H_T	0	0	0	0	0	0	V_R	0	0	0
s_5	0	-2	0	0	0	0	-1	0	H_T	0	0	0	0	0	0	V_R	0	0
s_6	0	0	-2	0	0	0	0	-1	0	H_T	0	0	0	0	0	0	V_R	0
s_7	0	0	-2	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	V_R
r_1	0	0	0	-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_2	0	0	0	-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_3	0	0	0	0	-2	0	0	0	0	0	0	0	0	0	0	0	0	0
r_4	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0	0	0	0
r_5	0	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0	0	0
r_6	0	0	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0	0
r_7	0	0	0	0	0	0	0	0	-2	0	0	0	0	0	0	0	0	0
r_8	0	0	0	0	0	0	0	0	0	-2	0	0	0	0	0	0	0	0

a_2 is directly impacted if service S_4 is delayed. To identify activities which are indirectly affected due to possible delay in a_2, in the row of a_2 obtain the column which have the value H_T (which is a_3), hence a_3 is indirectly affected, a_2 and a_3 are added to the Impact_List.

5.1 Analysis

The identification of root-causes and impact analysis is essentially the traversal of vertices and directed hyperedges in the hierarchical SBS. The algorithms for root-cause and impact-analysis make use of the relation matrix to identify the dependencies between entities of a SBS. The time complexity of root-cause analysis algorithm is O (n) where n is the number of resources a service depends on. The time complexity of impact-analysis algorithm is O (n) where n is the number of services in the SBS.

6. Conclusions and Future Work

In this study we have presented a model for SBS as a hierarchical structure consisting of business process layer, service composition layer and service infrastructure layer using hypergraphs. We model the horizontal dependencies within each layer and vertical dependencies across the hierarchical layer structure using hyperedges. Our approach overcomes the inability of simple graphs in modelling multi-level dependencies. The dependency information is stored in a relation matrix and can be effectively utilised for performing impact and root-cause analysis when anomalies are observed in a SBS.

In our approach we have represented inter-layer dependency information from two facets i.e. vertical time and vertical resource dependency. But the intra-layer dependency information is represented only from the temporal facet. As a part of our future work we would like to consider representing horizontal relationships from multiple facets between two entities.

7. References

1. Cardoso J. Business process control-flow complexity: Metric, evaluation and validation. International Journal of Web Service Research. 2008; 5(2):49–76.
2. Kazhamiakin R, Pistore M, Zengin A. Cross-layer adaptation and monitoring of service-based applications. Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops; 2010 Jan 1. p. 325-34.
3. Winkler M, Schill A. Towards Dependency Management in Service Compositions. In ICE-B 2009. p. 79-84.
4. Sensarma D, Sarma SS. A survey on different graph based anomaly detection techniques. Indian Journal of Science and Technology. 2015 Nov; 8(31). doi:10.17485/ijst/2015/v8i1/75197
5. Bodenstaff L, Wombacher A, Reichert M, Jaeger MC. Monitoring dependencies for slas: The mode4sla approach. IEEE International Conference on Services Computing (CC'08); 2008 Jul 7. p. 21-9.
6. Zhou Z, Bhiri S, Hauswirth M. Control and data dependencies in business processes based on semantic business activities. ACM Proceedings of the 10th International Conference on Information Integration and Web-Based Applications and Services; 2008 Nov 24. p. 257-63.
7. Sell C, Winkler M, Springer T, Schill A. Two dependency modeling approaches for business process adaptation. Proceedings of the 3rd International Conference on Knowledge Science, Engineering and Management LNCS 5914; 2009. p. 418-29.
8. Keller A, Blumenthal U, Kar G. Classification and computation of dependencies for distributed management. Proceedings of 5th IEEE Symposium on Computers and Communications (ISCC); 2000. p. 78-83.
9. Wang S, Capretz MA. A dependency impact analysis model for web services evolution. IEEE International Conference on Web Services (ICWS); 2009 Jul 6. p. 359-65.
10. Omer AM, Schill A. Web service composition using input/output dependency matrix. ACM Proceedings of the 3rd Workshop on Agent-Oriented Software Engineering Challenges for Ubiquitous and Pervasive Computing; 2009 Jul 13. p. 21-6.
11. Wang R, Zhang Y, Liu S, Wu L, Meng X. A dependency-aware hierarchical service model for saas and cloud services. IEEE International Conference on Services Computing (SCC); 2011 Jul 4. p. 480-7.
12. Pan Y, Wu L, Liu S, Meng X. A hypergraph partition based approach to dynamic deployment for service-oriented multi-tenant SaaS applications. Enterprise Interoperability. Berlin Heidelberg: Springer; 2012 Jan 1. p. 185-92.
13. Wu L, Pan Y, Liu S, Li Q. Construct SaaS applications from multi-abstract-level: Method and System. IEEE 7th China Grid Annual Conference (ChinaGrid); 2012 Sep 20. p. 107-14.
14. Zhao D, Liu S, Wu L, Wang R, Meng X. Hypergraph-based service dependency resolving and its applications. IEEE 9th International Conference on Services Computing (SCC); 2012 Jun 24. p. 106-13.
15. Chen G, Zhong N, Yao Y. A hypergraph model of granular computing. IEEE International Conference on Granular Computing (GrC); 2008 Aug 26. p. 130-5.

16. Yao Y. A unified framework of granular computing. In: Pedrycz W, Skowron A, Kreinovich V, editors. Handbook of Granular Computing. Wiley; 2008. p. 401-0.
17. Yao Y, Zhong N. Granular computing. Wiley Encyclopedia of Computer Science and Engineering. 2008.
18. Gallo G, Longo G, Pallottino S, Nguyen S. Directed hypergraphs and applications. Discrete Applied Mathematics. 1993 Apr; 42(2):177-201.
19. Karimi M, Esfahani FS, Noorafza N. Improving response time of web service composition based on QoS properties. Indian Journal of Science and Technology. 2015 Jun; 8(16).
20. Saralaya S, D'Souza R. Cross layer property verification with property sequence charts. IEEE International Conference on Soft-computing and Network Security (ICSNS); 2015 Feb; 25(27):547–52.
21. Luckham D. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison: Wesley Professional; 2002 May.