International Journal of Control and Automation
Vol. 11, No. 1 (2018), pp.25-42
http//dx.doi.org/10.14257/ijca.2018.11.1.03

Process Modeling for Analysis and Control: A Case Study of IT Services

Sabah Al-Fedaghi[1] and Yousef Alduwaisan[2]

[1]*Computer Engineering Department, Kuwait University*
[2]*Information Technology Department, Ministry of Defense, Kuwait*
[1]*sabah@alfedaghi.com*, [2]*ykalduwaisan@mod.gov.kw*

Abstract

This paper is concerned with the notion of processes and, specifically, with business (organization) processes. The goal is to capture these processes in a way that facilitates their analysis and control most effectively. Current approaches to process modeling have been criticized for their lack of execution environment in which events that cause change would be incorporated. Such information is important for analysis and control in the study of dynamic behavior of systems. The heterogeneity and multiplicity of current models have caused problems that reveal the need to model events. Accordingly, the problem emphasized in this paper is the incorporation of events into process modeling. The paper introduces a method for producing a single, integrated diagrammatic representation that uniformly incorporates structural and behavioral aspects into an underlying conceptual description. The viability of the model is demonstrated by applying it to services in a case study organization unit.

Keywords*: Events; process modeling; conceptual model; system description; control*

1. Introduction

History shows that, for a long time, humans have tried to study the nature and dynamics of the world by applying observations and reasoning, and typically framing investigations from a perspective we would today classify as *systemic* (an ordered and harmonious structure) and in terms of systemic *processes* (processes that connect events and mediate, enable, limit, or condition phenomena) [1].

This paper is about the notion of processes and, specifically, about *business* (organization) processes. The goal is to capture these processes in a way that facilitates their analysis and control most effectively.

Process modeling and especially Business Process Modeling is a very active field these days. The businesses are trying to find a way how to keep their control over their dynamically expanding organizations and IT departments are trying to help them to achieve this goal. [2]

1.1. Process and Business Process

Typically, a *process* is said to refer to a collection of *activities* that take input and create an output of value to a customer ([3] referring to [4]). A *process* may also be defined as "a set of logically interrelated activities within an organization the execution of which contributes in the achievement of the business objectives" [5]. According to Svatoš [2] (referencing the U.S. Government Accountability Office),

[A *process*] is a collection of related, structured *activities*—a chain of *events*—that produce a specific service or product for a particular customer or customers. In reality

Received (May 18, 2017), Review Result (December 4, 2017), Accepted (December 14, 2017)

ISSN: 2005-4297 IJCA

there are very few well-structured processes. Analysts have to first understand reality and produce a *conceptual* (*i.e.*, a cognitive framework for organizing complex phenomenon) *representation* (of captured part of reality) that can form a base for structuring down the involved *process*. Designers take relatively well structured *process*; turn it in a model that can be later on used through some *process execution engine.* [2]

Accordingly, in these sources, a *process* is defined in terms of *activities*. It is typically said that an *activity* is a purposeful human action. In systems literature, an *activity* is described as an *operation* of the system [6].

The term *business* is typically associated with process; *business* is interchangeable with *organization*, a term applicable to all sorts of group entities such as government agencies and departments, charities, mutuals and cooperatives, *etc.* [7]. A *business process* is a continuous series of organizational tasks (smallest units of work), undertaken for the purpose of creating output [8].

1.2. Process Modeling and Conceptualization

Moreover, the paper concentrates on *process modeling*. Process modeling aims to identify and capture the process already being performed in reality or to take already identified and captured processes and make them formal [2]. The type of modeling involved in this paper is a diagrammatic representation that captures the flow of things (to be defined later) between a system and its environment and among system components.

Business process models are developed to document, optimize, and automate business processes [8]. In business, a diagrammatic representation of a process is commonly called *notation* [7]. *Business Process Model* (BPM) refers to a structural representation, description, or diagram that defines a specified *flow of activities* in a particular business or organizational unit [7]. It is *conceptual* in the sense that it is a picture that describes a real-world domain but does not include any implementation (*e.g.*, technology) aspects. It serves as a guide for the subsequent systems design phase and development.

Conceptual process modeling is the first and most important step in the BPM lifecycle [9]. With the development of Service-Oriented Architecture and Business Process Management technology, process modeling serves as the foundational theory of service combination and business process technology [10]. "Business process modeling is the study of the design and execution of processes" [10].

[Conceptual modeling is] the activity of formally describing some aspects of the physical and social world around us for purposes of understanding and communication… The adequacy of a conceptual modelling notation rests on its contribution to the construction of models of reality that promote a common understanding of that reality among their human users. ([11] as referenced in [12])

1.3. Contemporary Approaches: The Problem of *Event* in Process Modeling

Several notations are used to model processes, *e.g.*, BPMN [13], EPC (Event Process Chain) [14], UML activity diagrams [15], XPDL [16], YAWL [17], and IDEF 3 [18]. Business Process Modeling Notation (BPMN) was developed particularly for modeling business processes. Zimmermann *et al.*, [19] advocate the need for a hybrid approach that combines elements of all the approaches, with a number of distinct, new elements.

Current process modeling languages have been heavily scrutinized and critiqued. Take for example BPMN, described as representing "high-level graphical representation of business processes easily understood by business analysts, and especially useful in communicating business requirements" [20]; however, according to Dijkman *et al.*, [21],

The mix of constructs found in BPMN makes it possible to obtain models with a range of semantic errors… [and] the static analysis of BPMN models is hindered by ambiguities in the standard specification and the complexity of the language. The fact that BPMN integrates constructs from graph-oriented process definition languages with features for

concurrent execution of multiple instances of a subprocess and exception handling, makes it challenging to provide a formal semantics of BPMN. Even more challenging is to define a semantics that can be used to analyze BPMN models.

In general, current approaches have been criticized based on the following (see [2]):

- Modeling a process means programming a process while the aim is to understand reality and model a part of it.
- They see every change (transitions, activities, tasks) as the product of a function. The only differentiation that can be found is between simple (atomic) and complex functions. In conceptual modeling there are only activities and all are complex functions. There are no "functions."

However, the crucial problem highlighted in this paper is the incorporation of events into process modeling.

Related to this problem is Svatoš [2] criticism of such process modeling languages in this context as follows (all italics in quotes have been added):

- "There is no *execution* environment – we are trying to capture a part of reality and we should be interested what *events* cause activity state changes."
- In process modeling, one can see plans for various scenarios that are either used or not, depending on whether a specific *even*t occurs or a specific state of affairs has become true.
- "A conceptual language is focused on reality analysis, understandable and usable by non-programmers, all activities are complex, *activities* occur in time and there are states an activity can be in, there are *events* that cause transition to specified activity state, there are duties and rights and there are complex decisions" [2].

Harvey [22] claims that "The last thing the world needs is another Process language," yet this paper proposes a diagrammatic language appropriate for process modeling. This language depicts a machine comprising five basic "stages of flow": creation, release, transfer, receiving, and processing of things. Such a language could play a central role in business process modeling, including modeling of dynamic behavior.

In this paper we propose modeling business processes with a new type of notation, in which events are depicted as machines through which flowthings (to be defined later) pass. Here we make the case that our flow-machine model better captures the substance and dynamism of a business process by incorporating features that are missing from contemporary approaches.

For the sake of a self-contained paper, the next section briefly reviews the model [23-30] that forms the foundation of the theoretical development in this paper. The model has been adapted to several applications; however, the example given here is a new contribution.

Section 3 illustrates the proposed diagrammatic language and contrasts it with another diagrammatic language, the Data Flow Diagram, also based on the notion of flow. Section 4 explores the notion of structural models and their dynamic behavior in order to provide a theoretical foundation for our case study. Section 4 presents the theoretical foundation for incorporation of events into the model. Section 5 applies the FM model to a case study.

2. Flowthing Machines

A flowthing machine (FM) is a model representing things that "flow," *i.e.*, *things* that are created, processed, released, transferred, and received. "Things that flow" include information, raw materials, money, and much more. A *thing* is defined as what can be created, released, transferred, received, or processed while flowing within and between

stages and machines. The five stages of flow are shown in Figure 1 depicting a *flow machine*. The figure is an extension of the input→process→output description of processes.

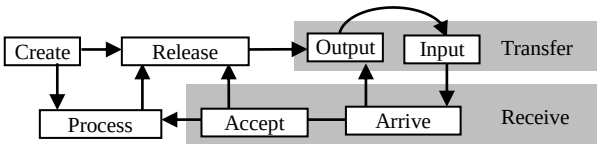

Figure 1. Flow Machine

The stages can be described as follows:

Arrive: A thing reaches a new machine.

Accept: A thing is permitted to enter a machine; if arriving flowthings are also always accepted, Arrive and Accept can be combined as a **Received** stage.

Process (change): The thing goes through some kind of transformation that changes its form.

Release: A thing is marked as ready to be transferred outside the machine.

Create: A new flowthing is born (created) in a machine.

Transfer: The flowthing is transported somewhere from/to outside the machine.

In general, a machine is conceived as an abstract machine that receives, processes, creates, releases, and transfers things. The stages in this machine are mutually exclusive. A thing in the Process stage cannot simultaneously be in the Create stage or the Release stage.

An additional stage of *Storage* can also be added to any machine to represent the storage of flowthings; however, storage is not an exclusive stage because there can be stored processed flowthings, stored created flowthings, *etc*.

FM also uses the following notions:

Spheres and subspheres: These are the environments of the machine. Multiple machines can exist in a sphere if needed. A sphere can be an entity (*e.g.*, a company, a customer), a location (a laboratory, a waiting room), a communication medium (a channel, a wire). A flow machine is a subsphere that embodies the flow; it itself has no subspheres. A sphere may sometimes be called a machine to denote the whole FM diagram.

Triggering: Triggering is the activation (denoted in FM diagrams by a **dashed arrow**) of a flow. It is a dependency among streams of flows. A flow is said to be triggered if it is created or activated by other flows.

A machine may not need to include all the stages; for example, an archiving machine might use only the stages Arrive, Accept, and Release with no Create stage. Multiple machines captured by FM can interact by triggering events related to other machines in those machines' spheres and stages.

3. FM Illustration: Contrast with Data Flow Diagrams

This section has two purposes: (i) to illustrate FM as a diagrammatic language, and (ii) to contrast it with another diagrammatic language, the Data Flow Diagram (DFD), which is also based on the notion of flow.

3.1. Contrast with Static Description

FM diagrams are meant to represent processes, similar to the purpose of DFDs [31]. "The data flow diagram is a modeling tool that allows us to picture a system as a network of functional processes, connected to one another by 'pipelines' and 'holding tanks' of data" [32]. A *process* in DFD is that "part of the system that transforms inputs into outputs; that is, it shows how one or more inputs are changed into outputs… the process sometimes describes who or what is carrying out the process" [32]. The Store notion in DFD is used as a necessary time-delayed storage area between two processes that occur at different times; however, processes in DFD imply data transformation (change) while in FM things are generated (Create stage), in addition to being processed .

Figure 2 shows a typical DFD for a small system given by Yourdon [32]. According to DeMarco [31] as quoted by Yourdon [32], "The notation is simple and unobtrusive and, in a sense, intuitively obvious. This is particularly important [for] — not the systems analyst, but the user!".

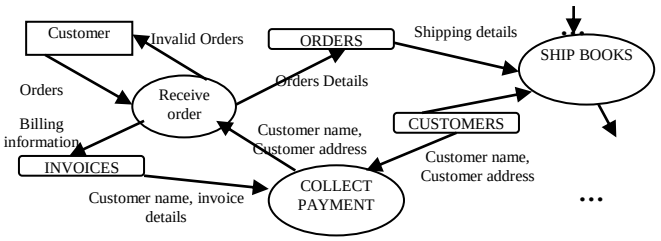

Figure 2. A Typical DFD (redrawn, partial from [32])

Figure 3 shows the corresponding FM representation. The customer creates an order (circle 1) that flows (2) to the purchasing department (3) of the supplier (4). There it is processed (5):

- If the order is invalid (6), it triggers the creation of a notification (7) that flows to the customer.
- If the order is valid, it triggers the creation of an invoice (8) that flows to the customer (9).

The invoice is processed (10) by the customer to trigger creation of a payment (11) that flows to the finance department of the supplier (13). There, the payment is processed to trigger the creation of a shipping order (14) that flows to the inventory department (15). In the inventory department, the shipping order is processed (16) to trigger release (17) of the purchased item to the customer (18).

Note that the FM diagram classifies "processes" into five generic kinds or "primitive processes" of Creation, Processing (changing), Receiving, Releasing, and Transferring. However, DFD has potentially an infinite number of processes corresponding to an infinite number of English verbs such as receive, respond, collect (payment), ship,

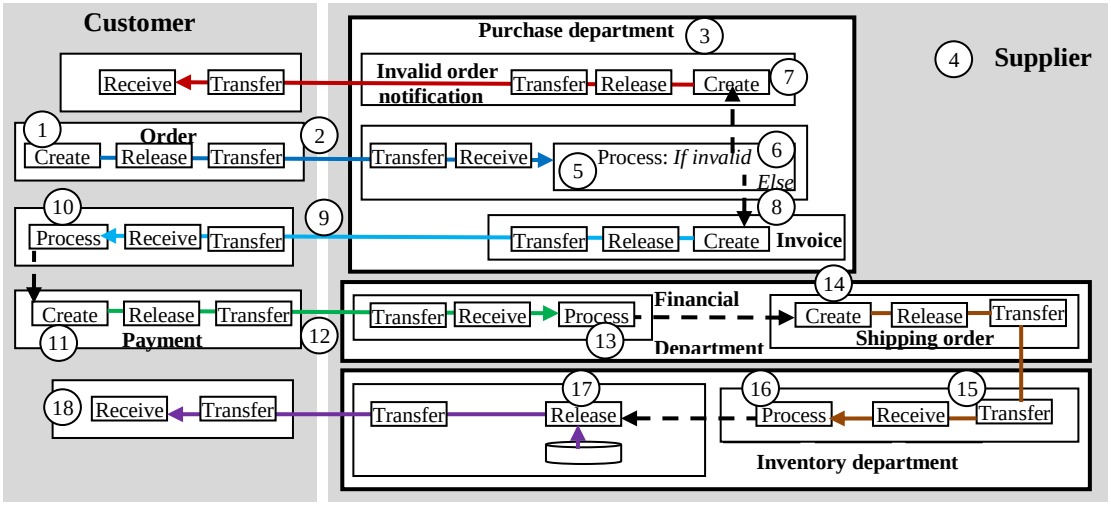

Figure 3. FM Representation of the Example

According to Yourdon [32], a typical DFD such as shown in Figure 2 "hardly needs to be explained at all; one can simply look at the diagram and understand it. The notation is simple and unobtrusive and, in a sense, intuitively obvious." The FM diagram can be described similarly. If needed, it is possible to produce the DFD by deleting the stages and making other minor additions as shown in Figure 4. The resulting DFD diagram is better than the original as it distinguishes among different flows (colored arrows); however, it is better to simplify the FM diagram by only deleting stages, as shown in Figure 5.

FM not only shows a part of the system that transforms inputs into outputs, but also opens the "black box" of the *process* in the input→process→output system. Thus, it is more "expressive" than the DFD by exposing the stages of Receive, Process, and Create.

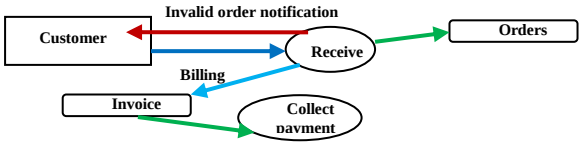

Figure 4. DFD Extracted from the FM Representation

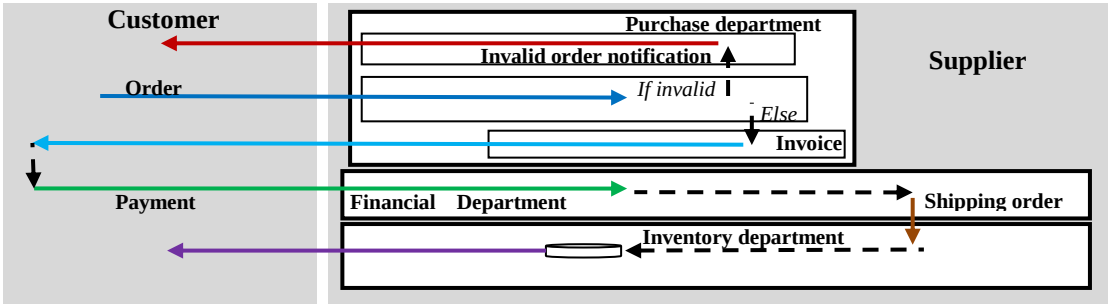

Figure 5. Simplification of the FM Representation by Deleting Stages

3.2. Contrasting Behaviors

According to Bækgaard [3], data flow diagrams are *event-less* in the sense that they are not based on an explicit notion of *events*, implying that non-event concepts are interpreted as if they were event concepts.

The development method Modern Structured Analysis uses essential events to trigger essential activities [33]. An essential activity is a coherent business activity that is modeled by a subset of a *data flow diagram*. The activity is triggered by the occurrence of a corresponding essential event. For example, an essential activity may represent the activity that is triggered when an order is received. [3] (Italics added).

FM explicitly models the notion of events, as will be demonstrated in the next section and applied in the case study to be introduced later.

4. FM as a Modeling Language

This section has two purposes:

(i) To demonstrate different features of FM modeling

(ii) To explore the notion of structural models and their dynamic behavior

The second purpose provides the theoretical foundation for our case study.

4.1. Structural (Static) FM Description

According to Niemann [34],

UML provides a mechanism for clarity and communication between peers and customers, without having to know the details of the system. Unfortunately, this does not yet give us deep look into the scenarios that make up these function points, and does not yet provide a mechanism for visually *executing* the model so that we can see all the behaviors of each function point. (Italics added)

Niemann [34] then analyzes the scenarios of a *Place Call* use case where each scenario is captured through a sequence diagram. Capturing all the "interesting" scenarios, and "formal semantics of the functional points of the system," provides "enough knowledge to begin system level decomposition to describe the high-level design."

How can this diagram be executed? If we look at [the Sequence Diagram], we have two entities or lifelines, which are receiving and injecting messages. This means that these lifelines have behavior, and through our UML model based tools, we can define behavior formally in one of two ways, using either Statecharts or Activity Diagrams... Statecharts and Activity Diagrams are the essence of executable models. [34]

In our case, we will approach this topic of "a mechanism for visually *executing* the model" by means of an example using Business Process Execution Language (BPEL) [35]. A client invokes the business process, specifying the name of an employee, a destination, a departure date, and a return date. The BPEL process checks the employee travel status and prices of a flight ticket with two airlines. The process then selects the lower price and returns the travel plan to the client. First, Krizevnik [35] builds a synchronous BPEL process, in which the client waits for the response, invoking two asynchronous airline Web Services. Figure 6 shows the FM representation corresponding to the given sequence diagram.

In Figure 6, the user first creates a request for travel (circle 1 in the figure) that flows to the BPEL process (2). Note that "port" (not mentioned in Krizevnik's [35] sequence diagram) in BPEL corresponds to conceptual Transfer in FM. The request is processed (3) to trigger creation of a request for travel status (4) that flows to the Employee Travel Status Web Service, where it is processed (5) to fetch the status (6) and send to BPEL process. There, the arrival of this employee status causes two triggers (7 and 8):

- Sending request for price 1 to airline 1 (9)
- Sending request for price 2 to airline 2 (10)

Each airline creates the required price (11 and 12) and the prices flow to the BPEL process (13 and 14), then to the client (15-16-17).

The resulting FM diagram reflects the overall "shell" of involved processes and can be enhanced by richer specification such as that embedded in the first common header pipe.

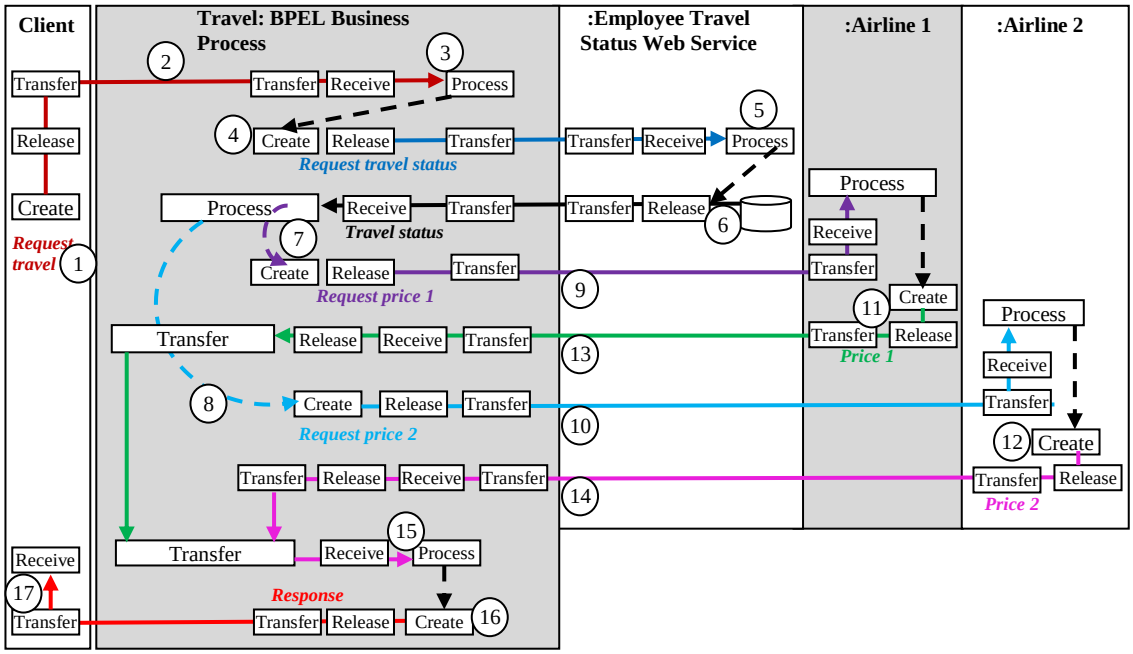

Figure 6. FM Representation of the Business Process Example

4.2. Behavior Description: Events, Instances, and Traces

FM modeling is a tool for capturing, analyzing, and developing activities. The chronology of activities can be extracted through orchestration of the sequence of events of the interacting processes. *Events* play a critical role in articulating interrelationships between processes. An event is a *thing* that can be created, processed, released, transferred, and received. Accordingly, choreography of the execution can emerge from the arrangement of events.

Modeling dynamic behavior occurs in a phase separate from the task of describing the anatomy map of Figure 6, representing an "events space" by a network of tracks and trajectories of recurring events throughout time. Events are the "trains" that run over these networks of tracks.

Figure 7 shows a sample event (type) that is not an elementary event (events can relate to each other in terms of inclusion, intersection, and so forth). It is selected because it is "meaningful" in the context of the travel example under consideration. Not every event is a meaningful event, just as not "every happening deserves the title of (historical) event" [36].

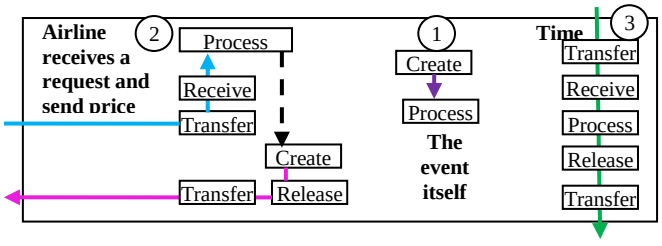

Figure 7. Composition of an Event

The top flow (circle 1) denotes the event itself. The event emerges (is created) and is processed (runs its course). What things pass through that event? They are the things participating in the flow of a request for a price and the response to that (2). These things

and their flows, triggering, and spheres form an instance of an execution. They go through time as an instance of the dynamic behavior of supplying a price. Finally, there is the time (3) of the event where processing of time means running its course. A trace is a sequence of events, as will be discussed next.

An event is a *happening* in a discontinuous period of time over its "space." Within this period and inside this "space shell," the *travel request* flows from the client to the BPEL process, triggering various flows according to the "script" (Figure 6) inside its conceptual space shell, during the time of the event. *Space* in this context reflects the specific conceptual location in the FM diagram—including spheres, stages, and tracks of flow—where an event *activates* the flows. Put simply, available spaces of events in the example under consideration are shown in Figure 6 or any of its sub-diagrams.

The behavior of the system is based on a schema such as Figure 6, where instances can be initiated and executed. Figure 6 can be instantiated as a synchronous or asynchronous trace of events.

The synchronous instances (Figure 8) comprise five "meaningful" events, as follows:

- Event 1: Client initializes a travel request that flows to the BPEL process; the client waits for the response.
- Event 2: This event starts upon receipt of the travel request from the client and embeds all operations taken by the BPEL process.
- Events 3 and 4: These are the events of sending requests for prices to two airlines.
- Event 5: This event starts upon receipt of responses from the airlines and ends with the client receiving the results.

Figure 9 shows the time line for this synchronous instance.

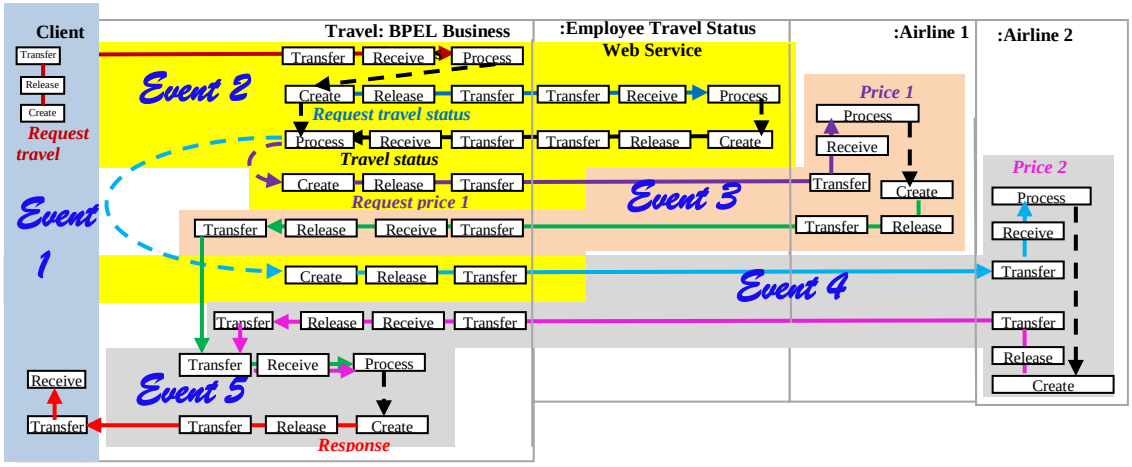

Figure 8. Synchronous Trace

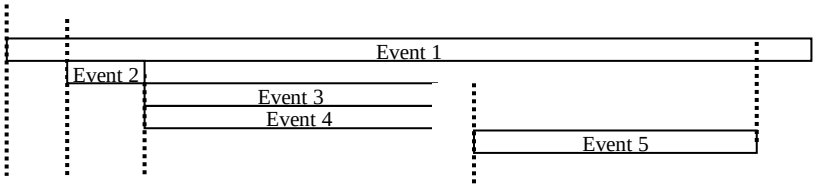

Figure 9. Time Line for Synchronous Trace

Krizevnik [35] then builds asynchronous invocations, as shown in Figure 10. Each event has a unique ID. Events can be characterized by various attributes, *e.g.*, an event can have a timestamp, correspond to an activity, be executed by a particular person, have associated costs, *etc*. Figure 11 shows the time line for this asynchronous instance.

The events in Figures 8–11 can be used to establish an overall control module, as will be demonstrated in the case study to be discussed in the paper.

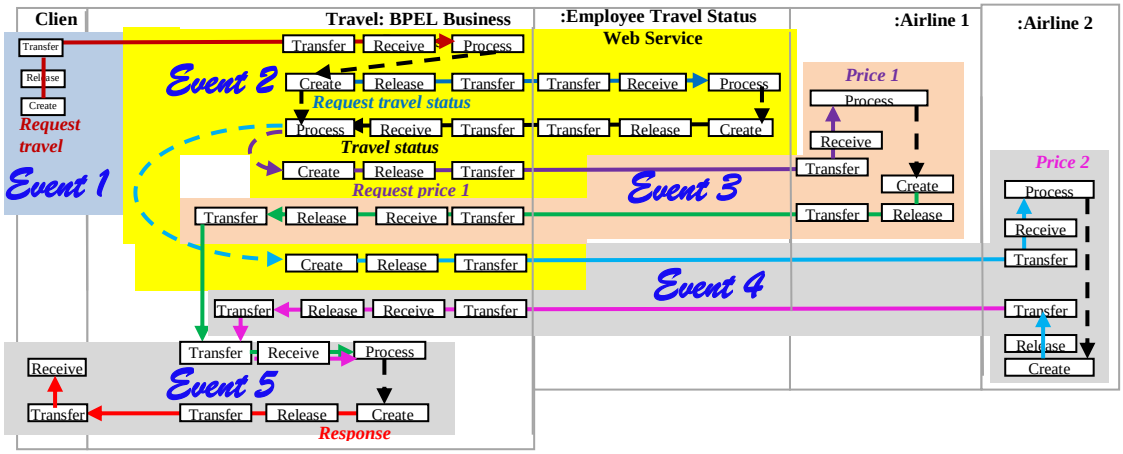

Figure 10. Asynchronous Trace

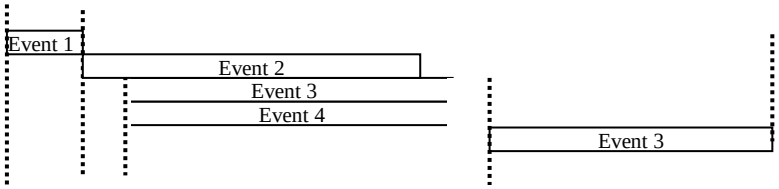

Figure 11. Time Line for Asynchronous Trace

5. A Study Case: IT Department

This section reports the results of applying FM as a process modeling tool to currently existing processes in the IT department of a Kuwait government ministry (where the second author is employed). Note that each process modeled in this section represents a cyber-physical system that integrates computation and physical processes, where users interact using both their digital identities and the physical interaction [37].

The IT department of the Kuwait Ministry of Defense has 160 employees assigned to diverse functions such as programming, systems analysis, network engineering, and working with contractor employees on hardware and software maintenance. The main hardware includes high-end servers including HP, HPEPro, Liant BL460c, and Gen9 blade servers which mostly run Microsoft Server 2012 R2. The main software used is Hyper-V, and the databases are in Oracle. The department is also responsible for maintaining the ministry's network built mostly of Cisco devices.

5.1. Employee Services

Every employee in the IT department is eligible and applies for various *services* and transactions within the department, including annual vacation, sick leave, overtime compensation, and personal data updating.

5.1.1. Current System

Handling such processes involves electronic and manual operations that have been captured in the FM representation shown in Figure 12. The diagram represents a conceptual static picture of processes as they are currently practiced.

Figure 12 includes an employee sphere, representing any employee (circle 1), the sphere of the State Audit Bureau (2, bottom of diagram), and the sphere of the IT department as a government organization unit. The Bureau is the government watchdog over bureaucratic processes in the state of Kuwait. For simplicity, various components of the IT department are not enclosed in a single box; rather, they are represented by shaded boxes in the figure. The process can be explained as follows:

In this service process, the employee, using his/her Web page, issues a request (circle 3) to download a certain type of application form, *e.g.*, annual vacation or sick leave. The information system of the IT department receives the request (4) and processes it to trigger (5) retrieval of the blank application form that flows to the employee (6). The employee prints a hard copy of the form (7), fills it in and signs it (8), and sends it to his/her team leader (9). The team leader processes and signs the application (10). The application then flows to the employee's supervisor, who approves it (11) and sends it to the Registration/Send/Receive section.

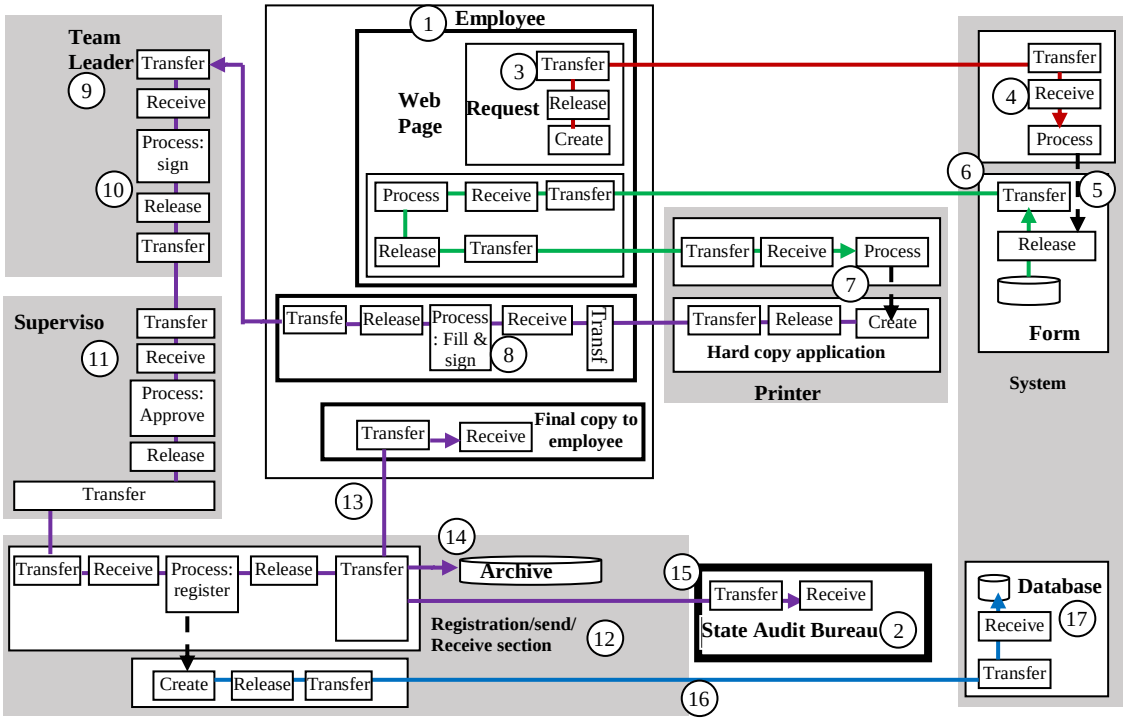

Figure 12. Employee Services

There, it is registered (12) and copies are sent to the employee (13), the department archive (14), and the State Audit Bureau (15). Additionally, a record is created (16) that is stored in the database of the system (17).

Currently, this process is understood implicitly. There is no documentation that describes the complete system. For example, a new employee is told orally to download the blank form, fill it in and sign it, and submit it to the team leader. He/she discovers more steps if something goes wrong in the process of the application.

Figure 12 forms the foundation for any type of development of the involved services. It is a static description much as would be offered from a philosophical perspective, a *form* (*i.e., having potential existence*) of what is being represented.

Figure 13 shows *an activation* of this form in terms of *events of change* or real world *happenings*. It reflects a process of initiating actions into the time-space continuum. Twelve "meaningful" *events* (from the functional point of view) make up such a process:

1. Requesting a blank form
2. Processing the request by the information system
3. Receiving the form
4. Making a hard copy
5. Filling in the form
6. Processing the form by the team leader
7. Processing the form by the supervisor
8. Processing the application by Registration/Send/Receive section
9. Application is sent to the Archive
10. Copy is sent to the Bureau
11. Copy is sent to the employee
12. Copy is sent to the database

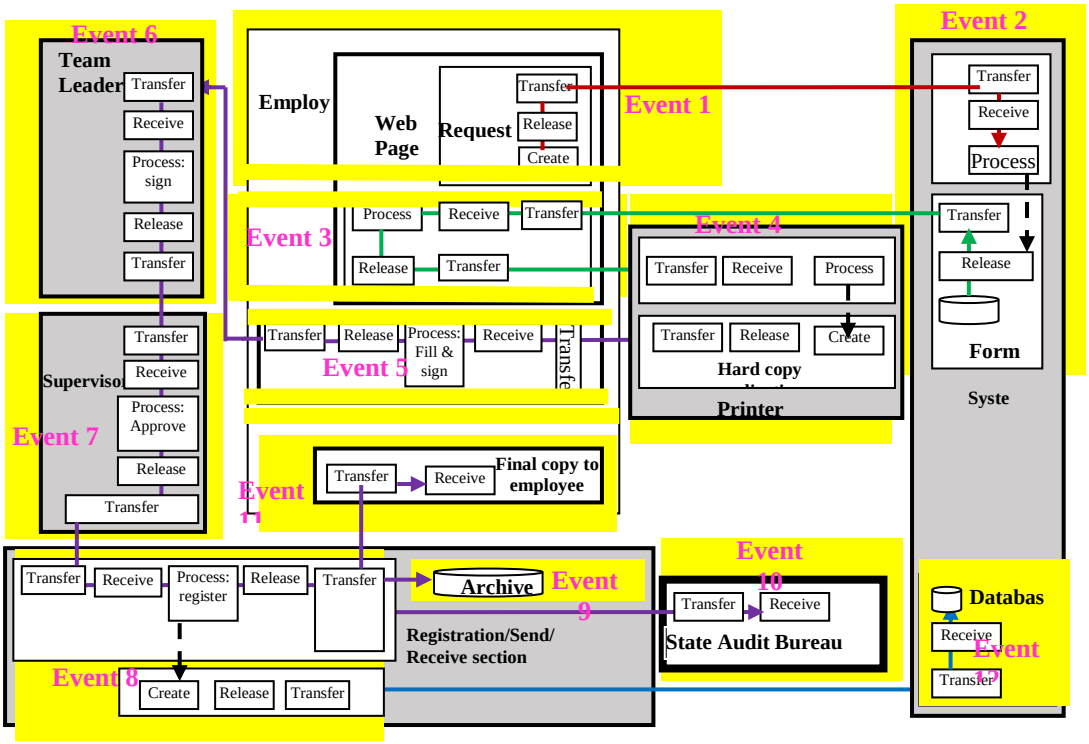

Figure 13. Meaningful Events

We can observe the passive role of the information system. It serves as a repository for blank forms and final applications. Note that the employee is a participant in four events in the list: 1, 3, 4, and 5.

5.1.2. Redesigning the System

It is possible to develop a control system that minimizes manual subprocesses, seen in event 1, and maximizes parallelism, seen in events 5–8.

We note that the FM representation provides a base for all types of development. Figure 14 shows a simplified view of the FM representation from which all stages have been removed. The *stick man* figure is borrowed from UML for illustration purposes.

This view can be used to provide an overview of the system, the same way block diagrams simplify the schemata of electrical and electronics circuity. It can also be used in

re-design of the system, as shown in Figure 15. From this last figure, a complete FM representation can be drawn as a first step in developing a new system.

As a simple indication of potential improvement in task processing with respect to the employee, he/she participates in two events: downloading and completing the application form online, and receiving final approval. Both do not involve any physical process, as opposed to, *e.g.*, having to make a hard copy in the current system).

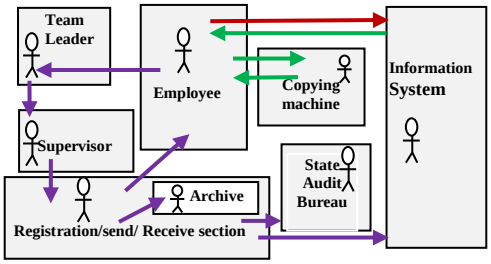

Figure 14. Simplification of Current System

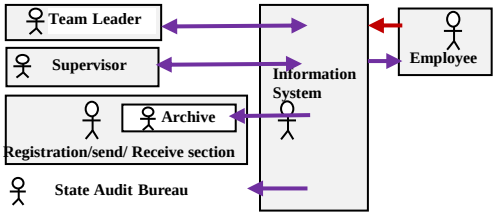

Figure 15. Redesign of Current System

5.2. Help Desk Control

The modern IT help desk process operates as a single-point-of-contact for clients (ITIL recommendations). Typically, flowcharts are used to describe its operations (*e.g.*, [38]). Figure 16 shows an FM representation of the current IT help desk in our case study.

An employee (circle 1 in Figure 16) files a complaint (2) that flows (3) to the *help desk* (4). There, it is processed (5) to trigger (6) examination of the list of available technicians in order to

- create a task (7-8)
- assign an available technician (9-10)

Note that creating a task (7) implies "filling in" details of the complaint (11) and name of the technician (10). Note also that the list of employees is shared between the help desk (4) and the technicians (12 – top right, brown in online version); thus in the diagram the sphere of the help desk (4) overlaps with that of the technicians.

The task flows to the assigned technician (14 – right side of diagram) where it is received (15) and processed (16). The processing (executing) of the task triggers processing of the complaint (17). This may involve requesting a device (18) for trouble-shooting and receiving it from the employee (19). The end of this procedure of processing the complaint leads to the creation and signing (20) of a report. The report must be signed by the employee (21); finally, the report is given to the desk help (22). The report is received (23) and processed (24) by the help desk. If the task is not complete, the following occurs:

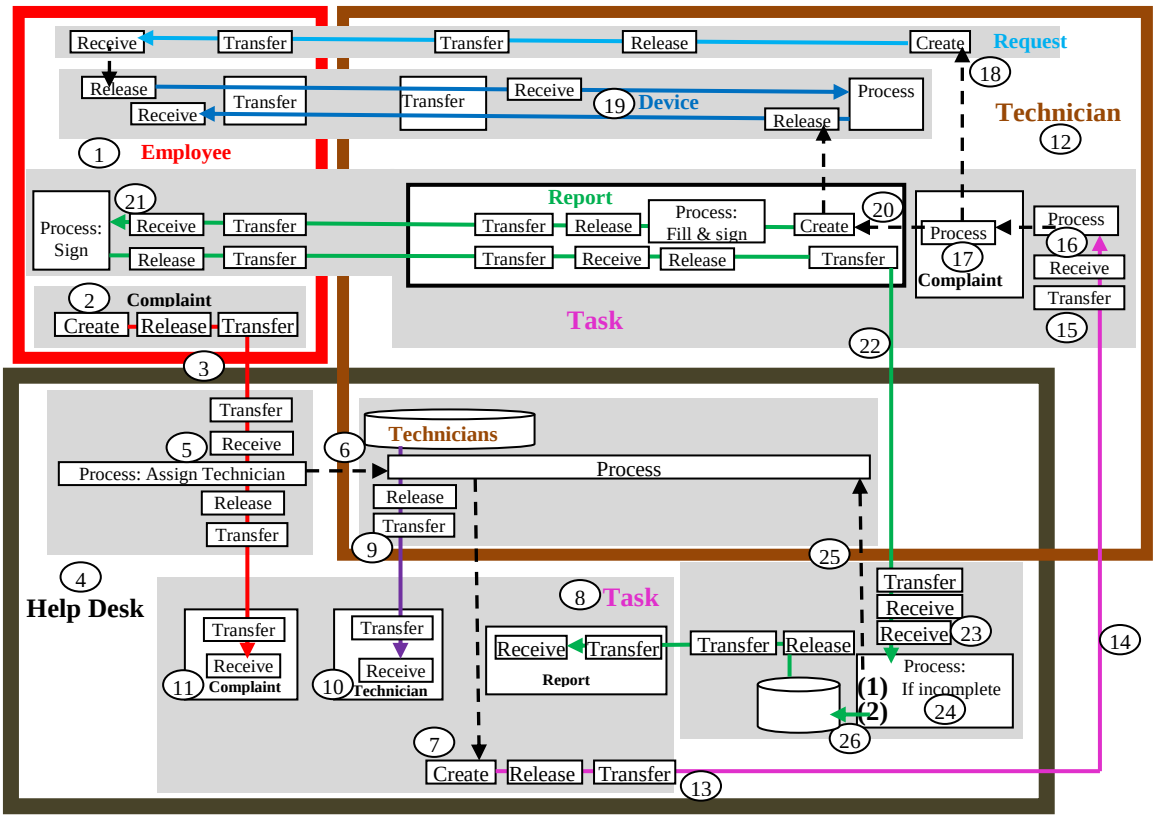

Figure 16. Help Desk Model

(i) The list of technicians is triggered to assign another technician (25), who flows to the unfinished task, and

(ii) The report is added to the task folder (26).

Accordingly, the revised task is assigned to the newly recruited technician, and the cycle of processing repeats.

From such a static FM description, we can identify all "meaningful" events, as shown in Figure 17:

V_1: Creation of a request for service

V_2: Help desk receives the request

V_3: Initiation of a service task that includes the following sub-events:

V_{3a}: Selecting a technician for the task

V_{3b}: Including a description of the complaint

V_4: Sending the task to the technician

V_5: The technician processes the task, which includes the sub-tasks

V_{5a}: Processing the complaint

V_{5b}: Receiving the serviced device

V_{5c}: Writing a report

V_6: Sending the report to help desk management

V_7: Processing the technician's report

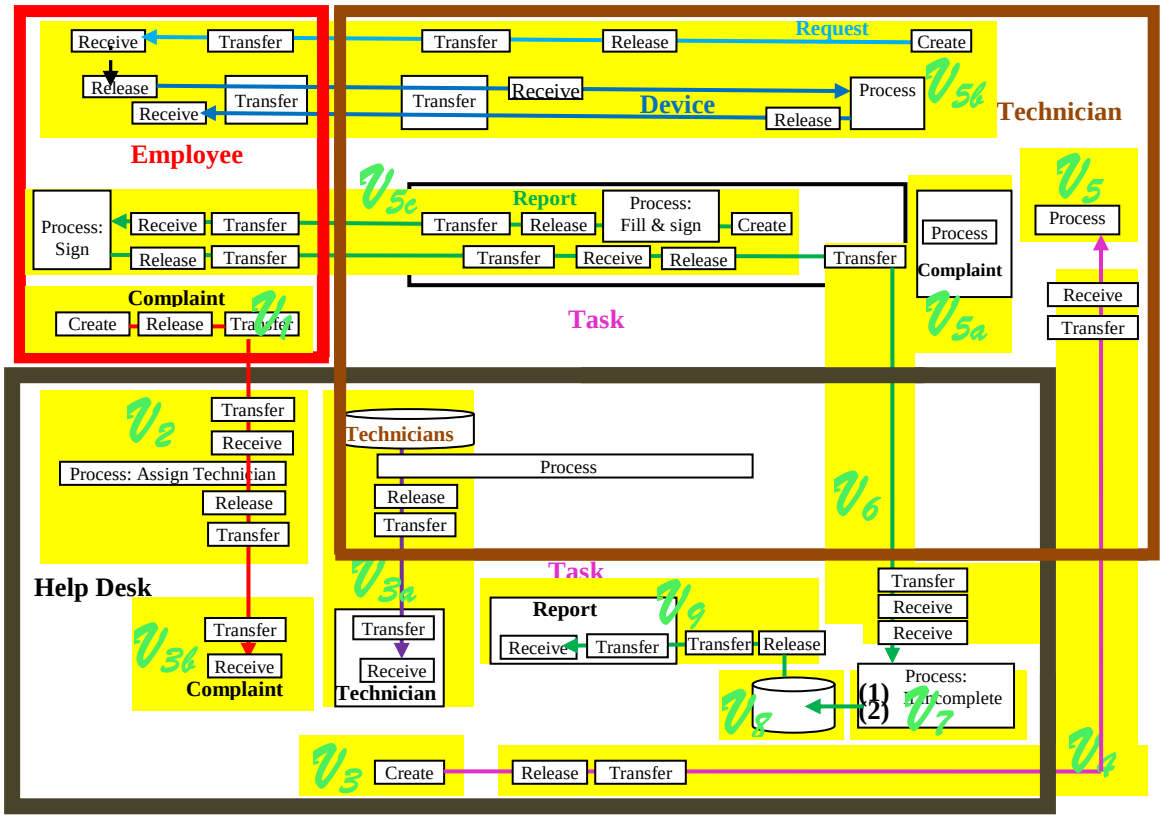

Figure 17. "Meaningful" Events in the Context of Management of the Help Disk

V_8: Archiving the report

V_9: If the task has not been finished, adding the report to reinitiation of the task

Figure 18 shows the control module of the help desk service.

Note that "meaningful" events refer to events of interest to the controlling unit of the help desk administration. Events can be selected according to requirements. For instance, suppose that VIP employees can use "emergency" services that bypass all red-tape requirements such as signing and reporting. Figure 19 displays events involved in this special case.

```
Service
{
V1              / Creation of a request/
V2              / receiving the request/
V3: {V3a, V3b}  /establishing a task/
Implement task {
V4              / Send task to the technician/
V5: V5a, {if needed V5b,    / processing by the technician: Bring the device if necessary/
          V5c}              / Perform the service then make a report/
V6              / Send the report/
V7              /Process the report/
V8              /Archive/
}
If the task is not finished   /a new version of the task /
{V3: {V3a, V3b, V9}
Process task
}
```

Figure 18. Help Desk Control

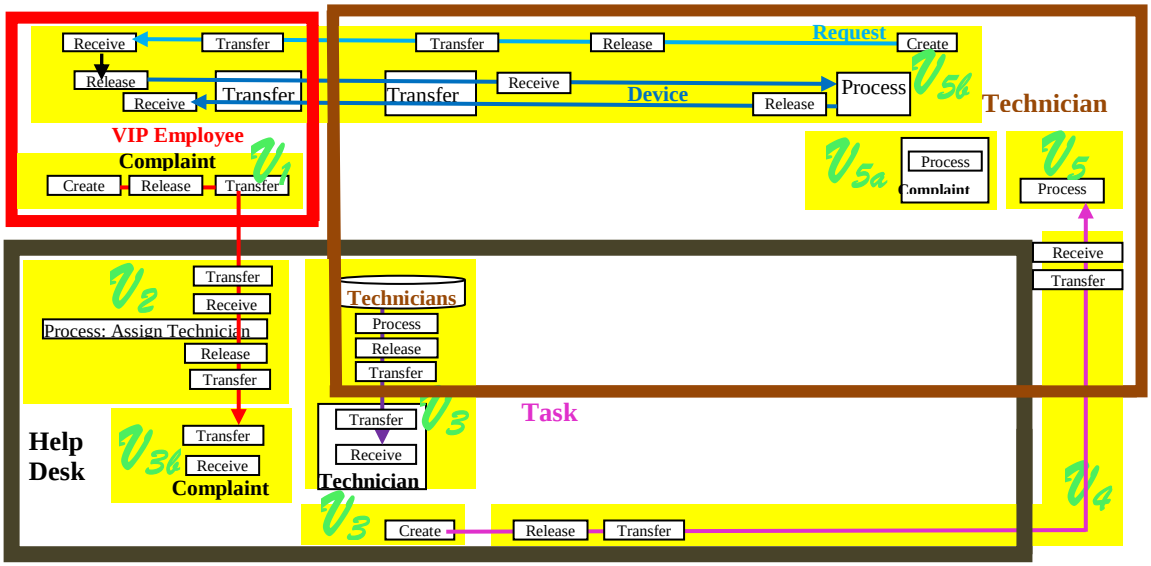

Figure 19. Events in Emergency Service for VIP Employee

6. Conclusion

This paper proposes a new method for modeling that can serve as an underlying framework for a single, integrated diagrammatic representation that uniformly incorporates the structural and behavioral aspects of a system. Modeling of events as a second level in modeling of structure and operations of a system while using the same language introduces an important development for integrating time into a conceptual representation. FM is viable as a modeling tool for including execution time events that are valuable for control modules in the system. It also provides a theoretical base for certain notions such as definition of *events*.

FM brings internal details up to the conceptual level. The complexity of the resulting diagram may introduce difficulties; however, some solutions have already been utilized in complex systems (*e.g.*, aircraft and high-rise building schemata) through multilevel simplifications. In FM, the details can be lumped together by omitting stages and unifying all types of flows in the model. Nevertheless, the underlying FM schema remains the reference for any further analysis and design.

Further research will work on implementing a management system for services in the case study environment. Many issues remain to be clarified; however, preliminary results demonstrate potential feasibility of the approach.

References

[1] Centre for Systems Philosophy, "The development and status of Systems Philosophy", **(2014)**. http://systemsphilosophy.org/history-and-development-of-systems-philosophy.html
[2] O. Svatoš, "Conceptual process modeling language: Regulative approach", Proceedings of the 9th Undergraduate and Graduate Students eConf. and 14th Business & Government Executive Meeting on Innovative Cross-border eRegion, Univ. of Maribor, **(2007)**.
[3] L. Bæ kgaard, "Event-based conceptual modeling", Business Process Management Journal, vol. 15, no. 4, **(2009)**, pp. 469-486. doi: 10.1108/14637150910975499.
[4] H. Hammer and D. McLeod, "Database description with SDM. A semantic database model", ACM Transactions of Database Systems, vol. 6, no. 3, **(1981)**, pp. 351-386.
[5] A. Tsalgatidou and S. Junginger, "Modelling in the re-engineering process", ACM SIGOIS Bulletin, Special Issue: Business Process Reengineering, vol. 16, no. 1, **(1995)**, pp. 17-24.
[6] D. A. Novikov, "Cybernetics, Control Philosophy and Control Methodology", chapter 2, Springer, **(2016)**. doi: 10.1007/978-3-319-27397-6.

[7] A. Chapman, "Business process modelling", Businessballs, **(2016)**. http://www.businessballs.com/business-process-modelling.htm
[8] J. Ziemann, T. Matheis and J. Freiheit, "Modelling of cross-organizational business processes: Current methods and standards", Enterprise Modelling and Information Systems Architectures, vol. 2, no. 2, **(2007)**, 23-31.
[9] W. M. P. van der Aalst, A. H. M. ter Hofstede and M. Weske, "Business process management: A survey", Lecture Notes in Computer Science, **(2003)**, pp. 1-12.
[10] G. He, G. Xue, S. Yao and Z. Wu, "Business process modeling: A survey", Proceedings of Annual Conference of China Institute of Communications, **(2010)**. file.scirp.org/pdf/6-1.43.pdf
[11] J. Mylopoulos, "Conceptual modeling and Telos", In P. Loucopoulos and R. Zicari, Editors, Conceptual modeling, databases, and CASE, chapter 2, Wiley, **(1992)**, pp. 49-68.
[12] G. Guizzardi, "Ontological Foundations for Structural Conceptual Models", CTIT PhD Thesis Series, No. 05-74. Enschede, the Netherlands, **(2005)**.
[13] Business Process Modeling Notation (BPMN) Specification, Final Specification, OMG, **(2006)**.
[14] A. Karhof, S. Jannaber, D. M. Riehle, O. Thomas, P. Delfmann and J. Becker, "On the de-facto standard of event-driven process chains: Reviewing EPC implementations in process modelling tools", In A. Oberweis and B. Reussner, Editors, Modellierung 2016, 2.–4. März 2016, Karlsruhe, GI-Edition: Lecture Notes in Informatics: vol. 254. Bonn, Germany: Köllen Druck+Verlag GmbH, **(2016)**, pp. 77–92).
[15] OMG, UML Resources, Object Management Group, **(1997-2016)**. http://www.uml.org/#home
[16] TIBCO, XPDL 2.1 Schema Overview, 2016 (Access). https://docs.tibco.com/pub/business-studio-bpm-edition/4.1.0/doc/html/GUID-6D561AC2-7865-4BE1-ABB7-BB5F3AA6D8FD.html
[17] N. Russell, A. H. M. ter Hofstede, W. M. P. van der Aalst and D. Edmond, "New YAWL: Achieving Comprehensive Patterns Support in Workflow for the Control-Flow, Data and Resource Perspectives", http://www.yawl-system.com/, **(2005)**.
[18] R. J. Mayer, C. P. Menzel, M. K. Painter, P. S. deWitte and T. B. B. Perakath, "IDEF3 Process Description Capture Method Report", Knowledge Based Systems, **(1995)**.
[19] O. Zimmermann, P. Krogdahl and C. Gee, "Elements of service oriented analysis and design", IBM Corporation, **(2004)**, https://www.ibm.com/developerworks/library/ws-soad1/index.html.
[20] R. K. L. Ko, S. S. G. Lee and E. W. Lee, "Business process management (BPM) standards: A survey", Business Process Management Journal, vol. 15, no. 5, **(2009)**.
[21] R. M. Dijkman, M. Dumas and C. Ouyang, "Formal Semantics and Automated Analysis of BPMN Process Models", Technical Report Preprint 5969, Queensland University of Technology, **(2007)**.
[22] M. Harvey, "Keeping BPM simple for business users: power users beware", BPTrends, January **(2006)**. http://mchavey.blogspot.com/2016/04/the-mike-havey-collection.html#!/2016/04/the-mike-havey-collection.html
[23] S. Al-Fedaghi, "Context-aware software systems: Toward a diagrammatic modeling foundation", Journal of Theoretical and Applied Information Technology, vol. 95, no. 4, **(2017)**.
[24] S. Al-Fedaghi, "Securing the security system", International Journal of Security and Its Applications, vol. 11, no. 3, **(2017)**, pp. 95-108.
[25] S. Al-Fedaghi and N. Aljallal, "Conceptual schematization of microcontroller and assembly language", International Journal of Software Engineering and Its Applications, vol. 8, no. 10, **(2014)**, pp. 179-190.
[26] S. Al-Fedaghi, "An alternative approach to multiple models: Application to control of a production cell", International Journal of Control and Automation, vol. 7, no. 4, **(2014)**.
[27] S. Al-Fedaghi, "System for a passenger-friendly airport: An alternative approach to high-level requirements specification", International Journal of Control and Automation, vol. 7, no. 2, **(2014)**.
[28] S. Al-Fedaghi, "Conceptualization of various and conflicting notions of Information", Informing Science, vol. 17, **(2014)**, pp. 295-308.
[29] S. Al-Fedaghi, A. Alsaqa, and Z. Fadel, "Conceptual model for communication", International Journal of Computer Science and Information Security, vol. 6, no. 2, **(2009)**.
[30] S. Al-Fedaghi and F. Mahdi, "Events classification in log audit", International Journal of Network Security & Its Applications, vol. 2, no. 2, **(2010)**.
[31] T. De Marco, "Structured Analysis and System Specification", Prentice-Hall, **(1978)**.
[32] E. Yourdon, "Just enough structured analysis", http://www.yourdon.com/, **(2006)**.
[33] E. Yourdon, "Modern Structured Analysis", Yourdon Press, **(1989)**. ISSN 0896-8470
[34] S. Niemann, "Executable Systems Design with UML 2.0", Telelogic, Andover, MA, **(2004)**. http://www.omg.org/news/whitepapers/Executable_System_Design_UML.pdf.
[35] M. Krizevnik, "Business processes with BPEL: An excerpt from WS-BPEL 2.0 for SOA Composite Applications with Oracle SOA Suite 11g", September **(2010)**. ttps://www.packtpub.com/books/content/business-processes-bpel
[36] C.-C. Tang, "Towards a real theory of event: A dialogue between Luhmann and Sewell", paper presented at Niklas Luhmann's Systems Theory in the World Society: A Resonance from Taiwan, Taipei: Institute of Sociology, Academia Sinica, **(2010)**.
[37] R. R. Rajkumar, I. Lee, L. Sha and J. Stankovic, "Cyber-physical systems: the next computing revolution", Proceedings of the 47th Design Automation Conference, **(2010)**, pp. 731–736.

[38] J. Ivarsson, "Quality Management for IT Support Services: A case study of an IT helpdesk service", Master of Science Thesis, Chalmers University of Technology, Gothenburg, Sweden, **(2013)**.

