Diagnosing Business Processes

Diana Borrego and María Teresa Gómez-López

16.1 Introduction

Most of the organizations related to industrial manufacturing are process-oriented. In order to facilitate their daily processes, enterprises use software, automating the execution order of their more frequent activities. Business Process Management Systems (BPMSs) have helped Industry to orchestrate their tasks and recover every used information, combining knowledge from information technology area and management science to improve operational business processes [2]. Business Process Models (BPMs) are formed of tasks combined by means of a workflow that lets the achievement of a particular business goal [21]. The BPMs can be enriched with rules to describe the values of the data that flows. For example, in the manufacturing process of an aerospace company, the process for aircraft assembly is able to be developed using a BPMS, because it is a frequent process where faults can turn up.

BPMs can be described using different perspectives, i.e., control flow, data flow, business rules, resources, and time. BPMs can or cannot work as expected, being necessary to diagnose them. The challenge is that business processes are designed to coordinate different elements related to an organization, then several of them are possible sources of a fault. For example, diagnosis can be related to the execution order of the activities, the input data introduced, the person who executes an activity, the behavior of an activity, etc.

Classical model-based diagnosis cannot be directly applied to BPMs, since they have special characteristics that are analyzed in this chapter. Aspects, such as the great quantity of data used, how the information is shared between different instances of a

D. Borrego (✉) · M. T. Gómez-López
Languages and Computer Systems Department, Universidad de Sevilla, Sevilla, Spain
e-mail: dianabn@us.es

M. T. Gómez-López
e-mail: maytegomez@us.es

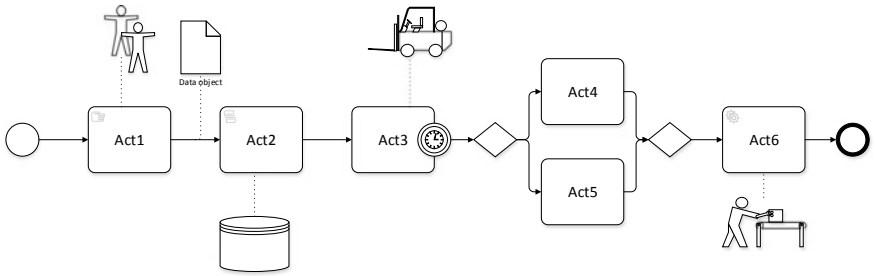

Fig. 16.1 Example of business process

process, or the heterogeneity of the used information or the shared resources, make necessary to tackle the diagnosis from others points of view.

An example of a general business process is depicted in Fig. 16.1, where a set of activities are combined to develop an objective in a company. The business process can include: who is executing an activity, what data objects are flowing in the instances, which resource is used in each activity, or the database whose data are read and written during the executions.

According to the business process life cycle (i.e., Design, Analysis, Configuration, Enactment and Evaluation [21]), Design, Analysis, and Evaluation are the most typical sources of faults, thereby where the diagnosis tends to be applied. These phases can be supported by a Process-Aware Information System (PAIS), that is, a software system that helps to manage and execute operational processes. PAISs include the possibility to model various perspectives, such as people, control flow, data layer or resources, and it is involved in every phase of the business processes.

The chapter is organized as follows: Sect. 16.2 introduces definitions about business processes and the main characteristics of BPMs to be diagnosed in their different phases. Section 16.3 explains the specific diagnosis techniques that can be used to tackle the challenge found in each phase of the business process life cycle. Finally, Sect. 14.5 concludes the chapter.

16.2 Problem Statement

Business Process Management (BPM) offers a set of mechanisms that facilitate the modeling of processes by nontechnical experts with a high level of automation [9]. Business Process Management Systems (BPMSs) are a set of software tools and methodologies focused on the management and recovery of business processes execution. The obtained information from the instantiations of the processes can be used to detect and diagnose incorrect behaviour or to improve the process executions. The use of this widespread recovery information enables enterprises to detect in a more efficient and effective way a malfunction in a system. Since BPMs combine various

systems related and choreographed, a more complete and complex diagnosis can be faced. BPM fits well with the necessities of factories [15] since every day they perform a set of well-defined processes to produce their products, where heterogeneous data are recovered.

16.2.1 Business Process Description

Business processes help organizations to describe and monitor their daily activities to be improved. The most relevant definitions are shown below:

Definition 16.1 (*Business Process*) A set of activities that are performed in coordination in an organization and technical environment. These activities jointly achieve a business goal [21].

The quality of a business process is able to be defined as both the capacity to produce quality outcomes or to produce the outcome in a quality way. Quality is usually measured by aspects, such as accuracy or security [14]. Verification and diagnosis of business processes help to ascertain the quality of the process, especially analyzing suitability and accuracy aspects in relation to the expected behaviour:

- **Accuracy**. The capability of a business process to provide the expected results with a certain degree of precision. The verification and diagnosis of business processes ensure obtaining the results in accordance with the expected behaviour, ensuring the attainment of the expected results.
- **Suitability**. The capacity to provide the appropriate solution to get the specified business goals.

Definition 16.2 (*Business Process Model (BPM)*) It is formed by a set of activities, a set of control flow gateways (AND, OR, XOR) that describes the relationship between the activities, and a set of Data Objects that flows in the business process [21]. BPMs can be activity oriented or artifact oriented.

Definition 16.3 (*Business Process Management*) It can be considered as an extension of classical Workflow Management (WFM) that includes concepts, methods, and techniques to support the design, administration, conformance, enactment, and analysis of business processes [21].

Definition 16.4 (*Business Process Management System*) It is a generic software system that is driven by explicit process representations to coordinate the enactment of business processes [21].

Definition 16.5 (*Business Data Constraint (BDC)*) They are a subset of business compliance rules that represent the compliance relation between the values of data during a business process instance. It is a numerical constraint that involves business process variables that can be related to the data flow or the stored information. BDCs can be associated as invariant to the business process or as pre- or post-conditions [13].

16.2.2 Business Process Phases

The whole business process life cycle includes various phases: In the **Design phase**, business processes are modeled, where typically livelock and deadlock analyses are performed [10] to avoid the incorrect synchronization of the project choreography. Also, data read and written must be analyzed, in order to avoid missing, redundant, and conflict data [19]. In the **Analysis phase**, validation and verification by simulating are performed. In order to cover the possible instances of a BPM, both control flow and data flow perspectives must be combined. A fully automated diagnosis of the correctness of BPMs implies to include the semantics of activities by means of business rules. What are the most relevant data to be monitored in a business process is not always a trivial problem. In the **Configuration phase**, the most appropriate points of monitoring are analyzed to obtain the level of diagnosability needed in each case. A correct BPM is not able to work correctly using incorrect input data. In the **Enactment and Evaluation phase**, processes are monitored and analyzed to evaluate and diagnose both BPMs (workflow and rules) [17] and the used data for each instance [13].

16.2.3 Model-Based Diagnosis and Business Processes

In order to diagnose if a system is working as expected, it is necessary to compare the designed model with the observations (Observational Model).

16.2.3.1 Model

The model in business processes tends to be formed of two parts: Activity-oriented business model and Business Data constraints.

Activity-oriented business model that tends to be described by using the standard BPMN [16] that permits the inclusion in the model of several artifacts to describe activities workflow, used data objects, choreography between processes, etc. The activity-oriented workflow describes the possible order of execution of the activities.

To include in the model how the variables can evolve during the execution of business process instances, business data constraints (Definition 16.5) describe the relation between the data values during the instantiation of a process. BDCs are presented as constraints with numerical variables (integer, float, and natural) by following the next grammar [13], where it is important to highlight that the variables participating in the constraints can come from the database or from the data flow:

```
BusinessDataConst := Atomic_Const BOOL_OP BusinessDataConst
 | Atomic_Const | 'NOT' Constraint
BOOL_OP := 'AND' | 'OR' | '→'
Atomic_Const := function PREDICATE function
```

```
function := Variable FUNCTION_SYMBOL function
  | Variable | Constant
Variable := Table'.'Attribute | Attribute | DataFlowVariable |
Constant
PREDICATE := '=' | '<' | '≤' | '>' | '≥'
FUNCTION_SYMBOL := '+' | '−' | '*' | '/'
```

BDCs are able to describe the pre- and post-conditions of the variables before and after an activity is executed. For example, {A_OutputSignal ≥ B_InputSignal + C_InputSignal AND A_OutputSignal ≤ C_InputSignal}.

16.2.3.2 Observational Model

In a Process-Aware Information System (PAIS), the information is typically stored in a relational database, and therefore the tuples of the tables compose the Observational Model (OM). Since the variables involved in a BDC have different origins, it is possible that attributes from various tables are related in a single BDC. The location of the data in various tables is due to the necessity to follow the *Normal Forms* defined in relational database theory. The normalization rules are designed to prevent update anomalies and data inconsistencies. Since data is stored in various tables, to ascertain the full tuple of values for an OM, a *denormalization process* needs to be carried out [13]. To understand how the OM can be extracted from a relational model, the example of Fig. 16.2 is used. It includes two entities (A and B) with a relation by means of a foreign key that relates each tuple of entity A with a tuple of entity B. If a join is executed between both entities, relation *Entity A&B* is obtained. This new relation represents three different OMs, where finally a diagnosis process can be executed. The different aspect, in this case, is that each OM is not independent of the others. For example, attributes *d* in first and second tuples come from a single value in the original entities. It means that if *d1* is an explanation of malfunction for a diagnosis, it has to cause a malfunction for the second tuple of the OM too. It is not always related to the value of the variable itself since, for example, the attribute *b* in the second and third tuples has the same value *b2*, but they represent two different variables that can take the same value. In the last case, the explanation of a malfunction in the second OM does not imply the same explanation for the last tuple.

16.2.4 Special Characteristics

Business processes have special characteristics that provoke the necessity of adapting classical diagnosis techniques. Both the great amount of data generated and the multisource heterogeneous information cause the need for adaptation of the classi-

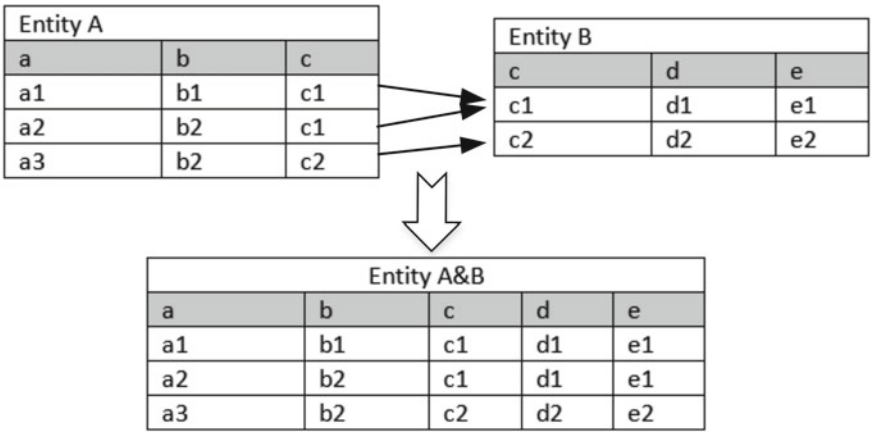

Fig. 16.2 Example of creation of OM

cal diagnosis for business processes. Some of the most relevant characteristics are explained in the following:

- **Shared data among different process executions**. Business processes tend to be modeled when a set of activities in a defined order are frequently repeated in a company. Each process execution is called instance. These instances use data that flow in the process or is stored in an external database. Derived from this external storage, the same data can be used in different instances, as explained in Sect. 16.2.3.2. It gives rise to when a value of a data is determined as incorrect in a diagnosis evaluation, it is necessary to take into account that the diagnosis explanation must be conformance with every instance where the data is involved, and if the data is incorrect or correct in an instance, it must be incorrect or correct for every instances where this data participates.
- **Different sources of fault**. Business processes are able to coordinate every participant in an organization into the same model. This has the advantage of integrating several possibilities in a diagnosis study, but also makes more complex the detection of the possible responsible for a malfunction. Several causes can be the origin: the involved persons, data, activities, resources, etc. This heterogeneity of possible faults makes more complex the isolation of the responsible for a malfunction.
- **Rules and structures combined**. Models in business processes are formed by the activities described in the *Business Process Model* of Definition 16.2, and the Business Data Constraints of Definition 16.5. This causes that, when a diagnosis is done, the whole process has not been analyzed, since it depends on the point of analysis in the business process instance.
- **Choreography between several processes**. Processes usually work in a coordinated way, although it brings about more complex models that can imply more faults. It is derived from several persons working and interacting in different parts of the models, increasing the possibilities of introducing malfunctions.

- **High changeability of business rules**. Business rules are frequently changed, since rules can change over time. This is because of compliance rules let business experts describe how the process works, not being necessary to modify the workflow.

16.3 Proposed Approach

The mentioned characteristics affect the diagnosis in different ways according to the involved phase. In following subsections, each phase is analyzed and how the diagnosis can be tackled derived from the special characteristics is explained.

16.3.1 Design

Only a reduced percentage of software projects succeed. The primary reason for these failures is poor conceptual modeling. To detect and avoid this, the aim of verification analysis methods is to check the correctness of BPMs, concerning the detection of syntax errors or violations in control flow and data flow perspectives.

16.3.1.1 Workflow

Most modeling and verification approaches in the literature only consider the control flow perspective of BPMs. These contributions typically deal with errors in the modeling of the flow, detecting deadlocks, livelocks (infinite loops), lack of synchronization, and dangling reference, most of them checking an important correctness criterion known as the soundness property, which guarantees proper termination of the business processes.

The most popular and effective notation for modeling business process at the conceptual level is the Business Process Modeling Notation (BPMN). It is a notation used to characterize the identification and the specification of the business processes. However, the general modeling includes the arbitrariness and lacks the strictness. For example, the congestion of business flows. If the congestion is not properly controlled, the software may not meet the requirements.

As explained in van der Aalst in [1], process chains are considered to be sound if they fulfill a series of conditions: (i) for every process A which is reachable from a start process, there exists a sequence leading from process A to the terminate process; (ii) the terminate process is the process that has no subsequent process; and (iii) for each process A, there is at least one occurrence sequence from the start process to the terminate process that involves process A.

In detail, the congestion can lead to the structural problem such as livelock. A livelock problem is an infinite execution of a process. In this problem, some of the

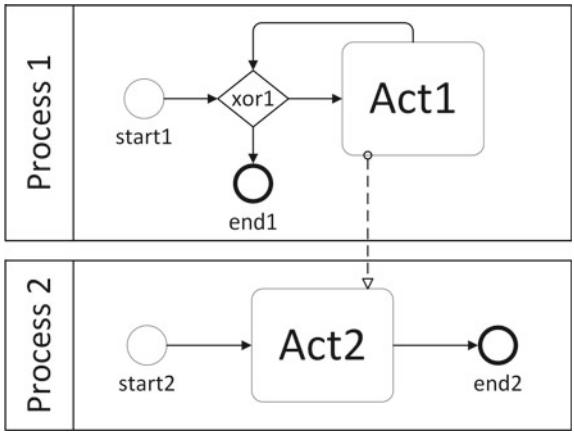

Fig. 16.3 Livelock and deadlock example

processes may run successfully but some of the processes trap in an endless loop of execution.

Likewise, deadlock takes place when a business process requires the execution of previous processes to be completed, but at least one of them is never executed. Such errors inhibit the BPM to work correctly.

As an example, Fig. 16.3 shows a simple BP model, where two processes collaborate by means of exchanging a message (dashed arrow). Depending on the evaluation of the exclusive gateway in Process 1, different traces can be obtained, which is possible to get stuck in a livelock or a deadlock. Such possible traces are

$T1 : (start1, start2) \rightarrow (xor1, Act2) \rightarrow (Act1, end2) \rightarrow (xor1) \rightarrow (end1) \Rightarrow$

$CORRECT\ TRACE$

$T2 : (start1, start2) \rightarrow (xor1, Act2) \rightarrow (end1, Act2) \rightarrow (Act2) \Rightarrow DEADLOCK$

$T3 : (start1, start2) \rightarrow (xor1, Act2) \rightarrow (Act1, end2) \rightarrow (xor1) \rightarrow (Act1) \rightarrow (xor1)$

$\Rightarrow LIVELOCK$

16.3.1.2 Data Flow

On the other hand, besides the structural verification regarding conflicts in the control flow perspective, when the data flow perspective is not correctly designed within the BPM, it can cause errors or conflicts, referred to as data anomalies in [18], classified into three types in [19] as given below:

1. Missing data. It takes place when a data item is accessed before it is initialized, for four different causes: (i) a data is never assigned an initial value; (ii) an activity which uses a data item is executed before the activity which initializes

it; (iii) the initialization and access are performed in parallel branches, thereby the data item may not have been initialized when it is read; and (iv) the data item is used but not initialized under certain workflow routing conditions.
2. Redundant data. A data item that does not contribute to the production of the final output data of the process is produced, causing inefficiency. The main cause is that a data item is produced by an activity, but never used by any activity executed after it in the process instance.
3. Conflicting data. There exist different versions of the same data item. It is impossible to decide which version of the data item should be considered.

In addition, it is necessary to consider verification of data flow errors, specifying variables read and written per activity, and even detailing the effect of each activity by means of pre- and post-conditions.

16.3.2 Analysis

The verification of BPMs allows to determine the correctness of business processes, because a business process cannot be correct, if: (i) one or more of its activities can never get executed for any possible value of the data input or (ii) there is not any possible process instance for a certain value of the input data. In these cases, despite the BPM can be correct regarding the control flow perspective, it is not correct from the perspective of the data flow.

In order to diagnose faults in data semantics, an effective means to express data constraints is to annotate activities in a BPM with pre- and post-conditions that specify the effect on the data state for each activity. Annotating process' activities with pre- and post-conditions facilitates compliance checking to ensure that business processes are properly designed. It is an effective means to capture such dependencies between control flow and data flow. Moreover, pre- and post-conditions can capture requirements on the data flow that any execution of the business process has to comply with.

This subsection presents an approach for diagnosing the correctness of the relationship between values in semantic BPMs, containing activities whose effects are formally specified using pre- and post-conditions. An activity can start if the execution of the BPM has reached the activity and its precondition is satisfied. Upon completion, the activity delivers data that satisfies its post-condition. An execution of the business process can reach an activity whose pre-condition is not satisfied. In that case, the execution gets stuck in the activity and fails.

To detect these errors, we distinguish between two different notions of correctness to diagnose such data flow errors [5]:

- May-correctness. A BPM is may-correct if every activity can be reached and executed at least once, so there is an execution in which the activity is done. The fulfillment of this correctness ensures that there is at least one valuation of the data inputs which makes an activity executable.

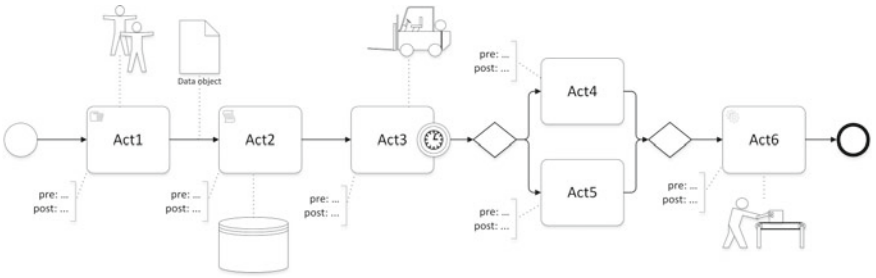

Fig. 16.4 Business process with annotated activities

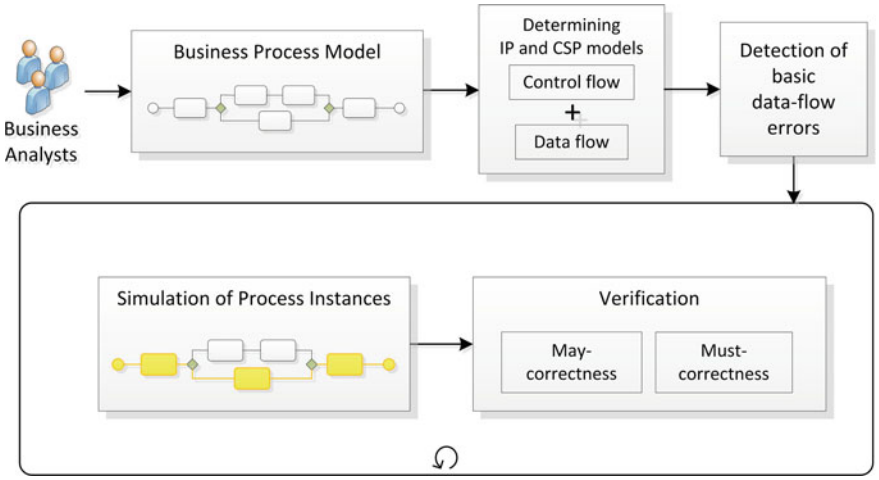

Fig. 16.5 Diagram of the verification process

- Must-correctness. A BPM is must-correct if every possible execution that reaches an activity satisfies the pre-condition of the activity. The fulfillment of this correctness ensures that there is no valuation of the data inputs which makes an activity non-executable.

To illustrate the idea, Fig. 16.4 shows a business process whose activities are annotated with pre- and post-conditions. Those conditions should be satisfied in accordance with the valuations of the data flowing and changing during the execution. As a concrete example, if activities *Act1* and *Act2* in Fig. 16.4 assign the valuation $\{a = 30, b = 100, c = 30, d = 200\}$, and the pre-condition of *Act3* is $pre(Act3) = 3 * a + b + c < d$, it would not be possible to execute *Act3* since $3 * 30 + 100 + 30 \not< 200$, so that the process would get stuck at *Act3*.

Diagram in Fig. 16.5 illustrates the diagnosis process, which is performed in the analysis phase, at design time, using artificial intelligence techniques to compute the execution instances allowed by a BPM. In detail:

1. The diagnosis process starts from BPMs with correct control flow definitions.
2. For diagnosis, the BPM is translated into two models: (i) an Integer Programming model (IP model), to determine the different instances of execution of the business process, and (ii) the pre- and post-conditions of the activities are modeled as constraints in a Constraint Satisfaction Problem (CSP), following a Backus–Naur Form (BNF) grammar in order to avoid any ambiguity.
3. Next, basic data flow errors are determined. These data flow errors at the basic level can cause unexpected process interruptions and should, therefore, be avoided. In detail:

   - Conflicts among postconditions: It is possible to find different postconditions that seem to be conflicting regarding the value of some variable. However, all of them could be satisfied since the value of a variable can be modified several times during the execution of a process, therefore making possible that those different valuations satisfy each postcondition as each activity is executed. In order to resolve conflicts among post-conditions, it becomes necessary to convert the variables and constraints of the CSP model into Static Single Assignment (SSA) form [3, 8]. As a result, in a CSP in SSA form, each variable is assigned a value by only one activity.
   - Missing data: When a variable is read by an activity, so referenced in its pre-condition, but not written in any preceding activity. To identify this kind of errors, the CSP is enriched with constraints that check the presence of at least a direct path from an activity writing a data and any activity reading it, so that the writing precedes the reading.
   - Conflicting data: When the same variable is written in two parallel activities. In that case, one activity overwrites the value of the variable written earlier by the other activity. The identification of this kind of errors implies the use of an algorithm to iterate over all activities writing a data, testing whether there exists any trace triggering any pair of them.

   Such data flow errors are at a more basic level than violations of must and may correctness. Therefore, these data flow errors need to be detected and resolved before may- and must-correctness can be diagnosed.
4. Finally, from the final obtained CSP, it is possible to simulate executions to check if the pre-condition of the final activity is satisfied in a scenario where all pre- and post-conditions of previous activities are satisfied, for both types of correctness:

   - May-correctness: There is at least one valuation of the data inputs which makes an activity executable.
   - Must-correctness: There is no valuation of the data inputs which makes an activity non-executable.

16.3.3 Configuration

Since there is no single entity which provides an overall view of the complete data flow model of a business process, it is desirable to improve the monitoring of business processes to be fully aware of possible deviations from expected behaviour.

Since the monitoring of a business process, and later diagnosis, is based on observations providing information about the behavior of the process, for the later diagnosis process to be successful, diagnosability analysis becomes a configuration time requirement. The diagnosability level of the business process depends upon the observations. If the diagnosis is based on few observations, or if observations are not allocated at the most convenient places, it is very difficult to distinguish which parts of the business process are failing. Both the number of observations and the location where they are performed enable the cause of the error to be precisely located. In general, diagnosis systems not only incorporate identification and isolation of faults, but also the monitoring.

The allocation of the test points is, therefore, a crucial task, since if the test points are not correctly allocated, then the observational model may be rendered useless for the isolation of faults, and hence the diagnosis process would fail to determine the activities which are not behaving as they were modeled.

Then, why not allocate test points at every possible location? Weighting the pros and cons, we would get a complete monitoring, but on the other hand, it would cause problems regarding confidentiality, privacy, and security, and it would also highly increase the necessary communications and cost.

So, test points should be located depending on certain requirements: (i) cost limitations, (ii) optimization of the diagnosis time, or (iii) expected diagnosability level.

As an example, for the business process in Fig. 16.6, the allocation of a test point as depicted gives rise to two clusters of activities (shadowed and non-shadowed activities, respectively).

Diagram in Fig. 16.7 shows the steps of the test points allocation process, which is performed in the configuration phase, using artificial intelligence techniques to compute the allocation depending on the chosen objective. In detail:

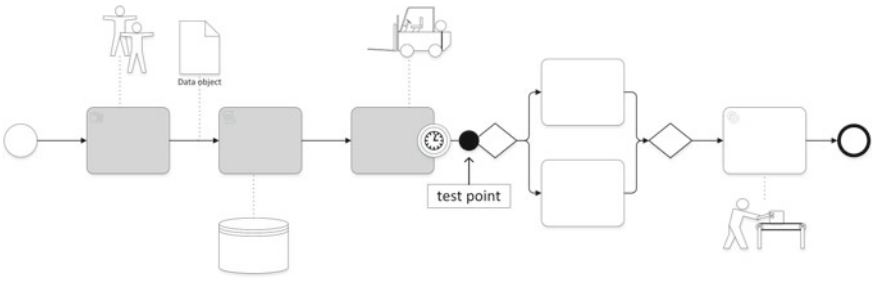

Fig. 16.6 Business process with test point

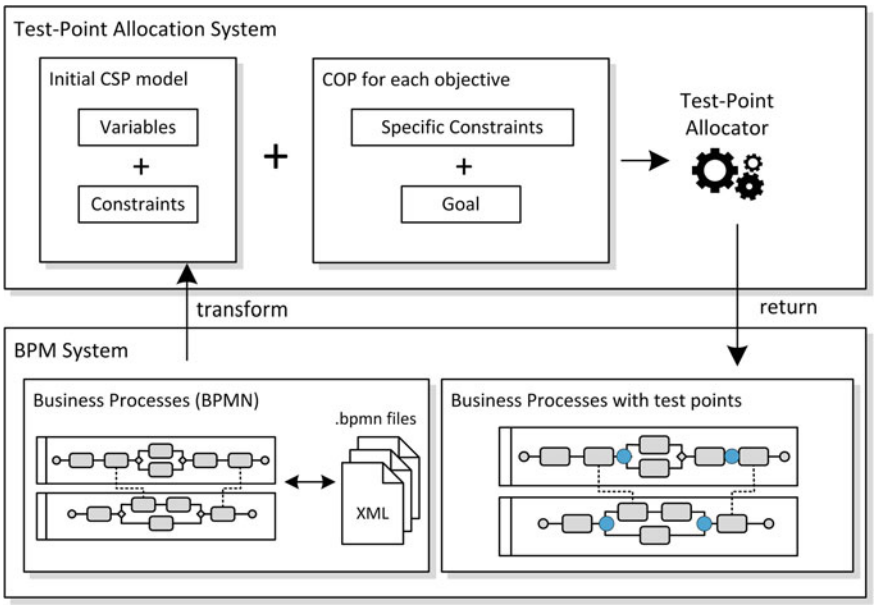

Fig. 16.7 Diagram of the process of allocation of test points

1. The process of allocation starts from BPMs with correct control flow and data flow definitions.
2. Then, the business process is modeled as an initial CSP, with no test points, so that all the activities belong to the same cluster.
3. Next, and according to the objective to achieve (cost, execution time, or diagnosability level) the initial CSP is enriched with specific constraints and goals for each objective, so that it becomes a COP. In detail, for each objective:
   - Cost limitations: It implies to limit the number of test points to allocate, with the goal of maximizing the number of clusters.
   - Minimization of the diagnosis process execution time: It begins with a fixed number of balanced clusters, with the goal of minimizing the number of test points to allocate.
   - Diagnosability level: From a maximum number of activities per cluster, the goal is to minimize the number of test points to allocate.
4. The allocation of test points is performed (test-point allocator) through solving the corresponding COP, getting different clusters of activities.

16.3.4 Enactment and Evaluation

A process correctly modeled and deployed can work incorrectly, since the use of incorrect data can cause improper working. Business Data Constraints can be used to validate data correctness during instantiation according to the Business Rules. Incorrect data can be included by users in the system or generated by activities during the process execution. The quantity of data can be huge and stored in external repositories, such as on a relational database. Therefore, the data involved in a business process instance that satisfy the BDCs are able to belong to the data flow (depending on the instance) and can be stored in a database (persistence layer).

Model-based diagnosis needs to compare the expected model and the observed model. Expected model is represented by a workflow that combines the activities and the Business Data Constraints. But the observational model must be extracted from external repositories to find the misalignment between the observed trances and the model [12]. Business processes usually are supported by relational databases that contain the traces for every instance. During a process execution, it is possible that a correctly designed model works incorrectly because of data. For this reason, it is necessary to analyze how to diagnose the process at different points at execution time. Figure 16.8 depicts the elements that are combined: BPMs, Business Data Constraints associated with activities or to the whole process, and external databases. BPM and Business Data Constraints conform the model of the system. The involved data stored in the database and flowing in the data flow are extracted to create the observational model as explained in [13].

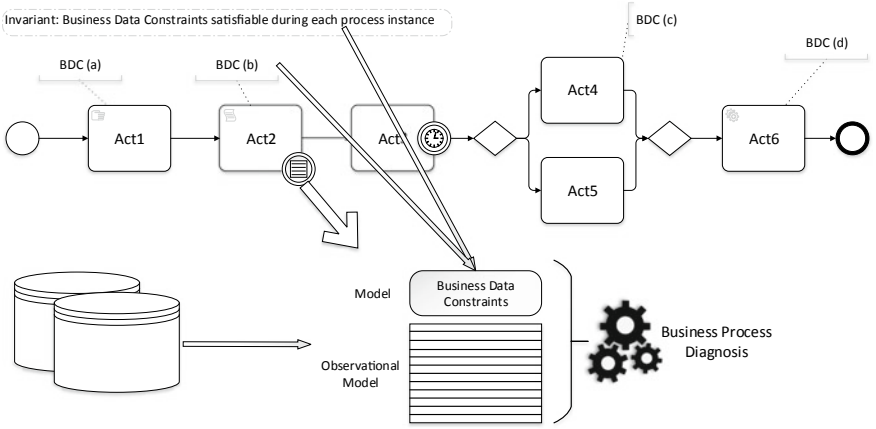

Fig. 16.8 Model and observational model of business process diagnosis at runtime

16.3.4.1 Constraint Programming to Find and Isolate Incorrect Values of Variables

As explained in the previous chapter, a Constraint Satisfaction Problem (CSP) consists of the assignment of a set of values to a set of variables which are involved in a set of constraints. A solution of a CSP is a total assignment satisfying all constraint. If no solution is found for a CSP, it means that the constraints are not satisfied between them, and the responsible component for this situation must be found. This is similar to the process of diagnosing in a system. For example, to find out the incorrect variable in a minimal diagnosis, we must eliminate some constraints, letting the CSP be satisfiable. To know which are the variables, finding an assignment which satisfies as many constraints as possible is a problem that can be solved using Max-CSP. The Maximal Constraint Satisfaction Problem (Max-CSP) consists of finding a total assignment which satisfies the maximum number of constraints. Max-CSP is an NP-hard problem and generally is more difficult to solve than the CSP problem [20]. Max-CSPs have already been used in model-based diagnosis [7] as a type of Constraint Optimization Problem (COP), as a CSP with an optimization function (maximization of minimization). Other different algorithms have also been proposed in order to improve the algorithmic determination of all minimal unsatisfiable subsets using notions of independence of constraints and incremental constraint solvers [4] and structural analysis [11].

Most times the problem is not related to incorrect restrictions nor a malfunction of the activities, very often the problem is due to incorrect input data. If we want to model a business process as a set of values for a set of business rules, where the error is the valuation of a variable for a tuple in a database, the constraints that can be relaxed to obtain a correct behavior will be the values of the variables. It means, to find the minimum number of variables that whether they are not instantiated, all the constraints are satisfiable.

Not all the variables can be incorrect for a direct human task, since some of them only depend on the values of input variables. For example, for the constraint $c = a + b$, a, and b are introduced by an user and c is derived from the values of a and b. It implies that if a or b are incorrect, then c will be incorrect too. For this situation, two types of variables are defined:

Definition 16.6 (*Input Variable*) Variable whose value is introduced by the user in an activity of a business process.

Definition 16.7 (*Derived Variable*) Variable whose value depends on the value of other variables (input or derived variables). It means that its value is not introduced as an input variable.

Once the input and derived variables are determined, depending on the business process, the goal of a diagnosis process is to detect the possible minimal incorrect variables set.

Definition 16.8 (*Possible Minimal Incorrect Variables Set*) Set of input variables ($\mathcal{S}$) that if they are not instantiated, the set of business rules are satisfiable. This set

is minimal iff there is not a subset $\mathcal{S}\sqcap\lfloor\mathcal{S} \subset \mathcal{S}$ where $\mathcal{S}\sqcap\lfloor\mathcal{S}$ is a possible minimal incorrect variables set.

The Max-CSP shown below is created to obtain this maximum set of variables in (1) with the value associated by the user. In order to obtain this minimal combination, reified constraints in (8) are used. A reified constraint relies on a variable that denotes its truth value. It is, therefore, necessary to add new variables to the CSP (in (2)), whose domain is reduced to values 0 (false value) and 1 (true value). The Max-CSP is

$$type\ InputVar_1, \ldots, InputVar_n \tag{16.1}$$

$$type\ InputVarBool_1, \ldots, InputVarBool_n \tag{16.2}$$

$$type\ DerivedVar_1, \ldots, DerivedVar_m \tag{16.3}$$

$$type\ varMaximize \tag{16.4}$$

$$InputVar_1 = [min\_domain_1, max\_domain_1] \tag{16.5}$$

$$\ldots$$

$$InputVar_n = [min\_domain_n, max\_domain_n]$$

$$DerivedVar_1 = [min\_domain\_derived_1, max\_domain\_derived_1] \tag{16.6}$$

$$\ldots$$

$$DerivedVar_m = [min\_domain\_derived_m, max\_domain\_derived_m]$$

$$Business\_Rule_1 \tag{16.7}$$

$$\ldots$$

$$Business\_Rule_k$$

$$Constraint_1 = (InputVarBool_1 = [InputVar_1\_Instance]) \tag{16.8}$$

$$\ldots$$

$$Constraint_n = (InputVarBool_n = [InputVar_n\_Instance])$$

$$varMaximize = Constraint_1 + \ldots + Constraint_n \tag{16.9}$$

$$maximize(varMaximize) \tag{16.10}$$

16.3.4.2 Hybrid Faults in Business Process Diagnosis

One of the challenges described in Sect. 16.2 is the continuous evolution of the rules in a business process. This is an advantage of flexibility in the system, but it is also a potential source of faults, because business constraints can be introduced incorrectly or being inconsistent according to the legacy data. Thereby, a great quantity of data and rules can be involved in a malfunction, then several possible diagnoses can be found. In order to reduce these possibilities, it is necessary to include weights associated with the various possibilities [6]. It is even more complex because of the high quantity of information and rules evolving in the processes at runtime. Derived from this complexity, a hybrid diagnosis needs to be performed regarding the likelihood of faults in data versus business rules. In order to achieve the correct diagnosis, it is fundamental to attain the best assumption concerning the degree of likelihood. For example, if the possible minimal explanation of a fault is that one data value is incorrect or one BDC is incorrect, which is the more likely minimal diagnosis? We could consider that data are more frequently introduced, and by more nonexpert user, then probably data is the incorrect element. But in the case that the possible explanation are four variables or one BDC? It could depend on the number of introduced input data. If 2,000 input data were introduced, 4 incorrect faults are a very low percentage, but if only 5 data have been introduced, it is more likely that the BDC is the incorrect element rather than 4 out of 5 values.

When both data and BDCs can be incorrect, the Max-CSP explained in the previous subsection must be adapted, since both variables and BDCs must be associated with a reified constraint that can be true or false. It is also necessary to include different likelihoods for each BDC, since it will depend on the number of variables associated to each one of them. For this case, a Min-CSP instead of a Max-CSP is needed.

In order to declare the **Variables of the problem**, a new variable is added to the Min-CSP for each variable obtained in BDC_j^i as explained in Sect. 16.3.4.1 following the syntax: *type* var_k^1, …, var_k^m.

Furthermore, in order to provide the Min-CSP with the ability to distinguish between different sources of faults (i.e., data and/or BDCs), also reified constraints are used. There are different reified constraints, some associated to each BDC (BDC_i) and some others to each BDC_i for each tuple j (BDC_i^j). The Boolean variable $rBDC_i$ will be true to denote the correctness of BDC_i for any tuple of values and rBDC_i^j for each different tuple. The Min-CSP has the following form:

//Boolean variables associated to reified constraint to ascertain the satisfiability of each BDC:

$$Bool\ rVar_k^1, \ldots, rVar_k^m \tag{16.11}$$

$$Bool\ rBDC_i, rBDC_i^1, \ldots, rBDC_i^n \tag{16.12}$$

//Reified constraints to represent the correctness of the variables assignment:

$$rVar_k^j = \neg(var_k^j = value_k^j) \tag{16.13}$$

//Variable to represent the reified BDCs for each tuple:

$$rBDC_i^1 = \neg(BusinessRule_i instantiated by tuple1) \tag{16.14}$$

$$\cdots$$

$$rBDC_i^n = \neg(BusinessRule_i instantiated by tuplen) \tag{16.15}$$

The reified variables are assigned to negated constraints since the objective function is to minimize the number of elements with incorrect behavior (noncompliant). To know when a fault in a BDC is less likely than in a data value, it is necessary to include in the Min-CSP the following BDCs:

$$rBDC_i = rBDC_i^1 + \cdots + rBDC_i^n = \sum_j^n rBDC_i^j \geq minLik_i \tag{16.16}$$

These constraints incorporate the likelihood concept into the CSP, by using the parameter minLik$_i$.

Definition 16.9 ($minLik_i$) This is a parameter that represents the minimum number of faults (noncompliant instances of a BDC$_i$) that is determined as the threshold to represent that there is a malfunction in a BDC$_i$ [6].

This value depends on the number of data related to each BDC. For example, if there are 5 tuples where BDC$_i$ is involved, $minLik_i$ can take a value between 1 and 5. If at least the $minLik_i$ threshold number of instances is not satisfiable, then BDC$_i$ is considered as a part of the minimal diagnosis. How the values of each $minLik_i$ is determined is detailed in [6].

The **Objective function** to minimize is defined as follows:

$$minimize(rVar_k^1 + \cdots + rVar_k^m + \cdots + rBDC_1 * minLik_1 + \cdots + rBDC_q * minLik_q) \tag{16.17}$$

Each rBDC$_i$ has a weighting that multiplies it that is proportional to each parameter $minLik_i$. This objective implies finding the minimal hybrid diagnosis.

Finally, it is important to include in the Min-CSP the following constraint, one for each BDC$_i$:

$$(rBDC_i^1 + \cdots + rBDC_i^n = 0) \vee (rBDC_i^1 + \cdots + rBDC_i^n \geq minLik_i) \tag{16.18}$$

Previous constraint contemplates two scenarios: (i) $\sum_j^n$rBDC$_i^j$ is equal to 0 being BDC$_i$ correct for every tuple of data or (ii) $\sum_j^n$rBDC$_i^j$ is equal to or greater than minLik$_i$, representing that the BDC$_i$ has a defect that is observed for the inconsistency of some (depending on the minLik) tuples. The inclusion of $\geq$ avoids intermediate values between 0 and minLik$_i$. This constraint strengths that a BDC$_i$ is incorrect iff another explanation will imply that more than MinLik input data were incorrect.

16.4 Conclusions

Business Process Models facilitate the description, implementation, and deployment of the activities of an organization. As other systems, both design and execution can be incorrect, being necessary the use of diagnosis techniques. Since industrial business processes have special characteristics, classic model-based diagnosis must be adapted to be applicable in this context. In this chapter, how the special characteristics can affect the most relevant business process phases have been presented, at Design, Analysis, Configuration, and Enactment and Evaluation phases. Regarding the Design and Analysis phases, both workflow and data management must be included in the business process diagnosis. In relation to the configuration phase, it is relevant to decide where a business process can be monitored to obtain the wished level of diagnosability. During enactment and evaluation phase, the model and data must be combined to detect incorrect behaviors at instantiation time.

Acknowledgements This work has been partially funded by the Ministry of Science and Technology of Spain (TIN2015-63502-C3-2-R) and the European Regional Development Fund (ERDF /FEDER). We would like to thank SACU of the University of Seville for the valuable information that has contributed toward the development of the ideas in this chapter.

References

1. van der Aalst, W.: Formalization and verification of event-driven process chains. Inf. Softw. Technol. **41**(10), 639–650 (1999). https://doi.org/10.1016/S0950-5849(99)00016-6
2. van der Aalst, W.M.P.: Process-aware information systems: design, enactment, and analysis. Wiley Encyclopedia of Computer Science and Engineering. Wiley, New Jersey (2008)
3. Alpern, B., Wegman, M.N., Zadeck, F.K.: Detecting equality of variables in programs. In: Proceedings of the 15th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp. 1–11. San Diego, California (1988)
4. de la Banda, M.G., Stuckey, P.J., Wazny, J.: Finding all minimal unsatisfiable subsets. In: PPDP '03: Proceedings of the 5th ACM SIGPLAN International Conference on Principles and Practice of Declaritive Programming, pp. 32–43. ACM Press (2003)
5. Borrego, D., Eshuis, R., Gómez-López, M.T.: Diagnosing correctness of semantic workflow models. Data Knowl. Eng. **87**, 167–184 (2013). https://doi.org/10.1016/j.datak.2013.04.008
6. Ceballos, R., Borrego, D., Gómez-López, M.T., Gasca, R.M.: Hybrid diagnosis applied to multiple instances in business processes. In: Enterprise, Business-Process and Information Systems Modeling - 17th International Conference, BPMDS 2016, 21st International Conference, EMMSAD 2016, Held at CAiSE 2016, Ljubljana, Slovenia, 13–14 June 2016, Proceedings, pp. 212–227 (2016)
7. Ceballos, R., Gasca, R.M., Valle, C.D., Toro, M.: Max-CSP approach for software diagnosis. In: IBERAMIA, pp. 172–181 (2002)
8. Cytron, R., Ferrante, J., Rosen, B.K., Wegman, M.N., Zadeck, F.K.: Efficiently computing static single assignment form and the control dependence graph. ACM Trans. Program. Lang. Syst. **13**, 451–490 (1991)
9. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Berlin (2013). https://doi.org/10.1007/978-3-642-33143-5

10. Falcioni, D., Polini, A., Polzonetti, A., Re, B.: Livelock and deadlock detection for pa inter-organizational business processes. In: Kő, A., Leitner, C., Leitold, H., Prosser, A. (eds.) Advancing Democracy, Government and Governance (2012)
11. Gasca, R.M., Valle, C.D., Gómez-López, M.T., Ceballos, R.: Nmus: Structural analysis for improving the derivation of all muses in overconstrained numeric csps. In: CAEPIA, pp. 160–169 (2007)
12. Gómez-López, M.T., Borrego, D., Carmona, J., Gasca, R.M.: Computing alignments with constraint programming: The acyclic case. In: Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data 2016 Satellite event of the conferences: 37th International Conference on Application and Theory of Petri Nets and Concurrency Petri Nets 2016 and 16th International Conference on Application of Concurrency to System Design ACSD 2016, Torun, Poland, 20–21 June 2016, pp. 96–110 (2016). http://ceur-ws.org/Vol-1592/paper07.pdf
13. Gómez-López, M.T., Gasca, R.M., Pérez-Álvarez, J.M.: Compliance validation and diagnosis of business data constraints in business processes at runtime. Inf. Syst. **48**, 26–43 (2015). https://doi.org/10.1016/j.is.2014.07.007
14. Heravizadeh, M., Mendling, J., Rosemann, M.: Dimensions of business processes quality (QoBP). In: Business Process Management Workshops, BPM 2008 International Workshops, Milano, Italy, 1–4 September 2008, pp. 80–91 (2008). https://doi.org/10.1007/978-3-642-00328-8_8
15. Kalpic, B., Bernus, P.: Business process modelling in industry - the powerful tool in enterprise management. Comput. Ind. **47**(3), 299–318 (2002). https://doi.org/10.1016/S0166-3615(01)00151-8
16. OMG: Object Management Group, Business Process Model and Notation (BPMN) Version 2.0. OMG Standard (2011)
17. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008). https://doi.org/10.1016/j.is.2007.07.001
18. Sun, S.X., Zhao, J.L., Sheng, O.R.L.: Data flow modeling and verification in business process management. In: AMCIS Association for Information Systems (2004)
19. Sun, S.X., Zhao, J.L., Nunamaker, J.F., Sheng, O.R.L.: Formulating the data-flow perspective for business process management. Inf. Syst. Res. **17**(4), 374–391 (2006)
20. Wallace, R.J.: Directed arc consistency preprocessing. In: Constraint Processing. Selected Papers, pp. 121–137. Springer, London, UK (1995)
21. Weske, M.: Business Process Management: Concepts, Languages. Architectures. Springer-Verlag New York Inc., Secaucus, NJ, USA (2007)