Expert Systems With Applications 183 (2021) 115212

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# Multi-criteria decision analysis for non-conformance diagnosis: A priority-based strategy combining data and business rules

Rafael Ceballos *, Diana Borrego, María Teresa Gómez-López, Rafael M. Gasca

*Department of Computer Science, University of Seville, Spain*

ARTICLE INFO

*Keywords:*
Multi-criteria decision analysis
Model-based diagnosis
Analytic hierarchy process
Business rule conformance
Constraint programming

ABSTRACT

Business process analytics and verification have become a major challenge for companies, especially when process data is stored across different systems. It is important to ensure Business Process Compliance in both data-flow perspectives and business rules that govern the organisation. In the verification of data-flow accuracy, the conformance of data to business rules is a key element, since essential to fulfil policies and statements that govern corporate behaviour. The inclusion of business rules in an existing and already deployed process, which therefore already counts on stored data, requires the checking of business rules against data to guarantee compliance. If inconsistency is detected then the source of the problem should be determined, by discerning whether it is due to an erroneous rule or to erroneous data. To automate this, a diagnosis methodology following the incorporation of business rules is proposed, which simultaneously combines business rules and data produced during the execution of the company processes. Due to the high number of possible explanations of faults (data and/or business rules), the likelihood of faults has been included to propose an ordered list. In order to reduce these possibilities, we rely on the ranking calculated by means of an AHP (Analytic Hierarchy Process) and incorporate the experience described by users and/or experts. The methodology proposed is based on the Constraint Programming paradigm which is evaluated using a real example. .

## 1. Introduction

Conformance can be understood as how well a system meets certain specified policies or standards. Organisational systems are becoming more and more complex, since information processes use knowledge-, service-, and cloud-based systems, thereby becoming the foundation of Big Data environments (El-Qurna, Yahyaoui, & Almulla, 2017). Conformance analysis implies analysing systems that run and support processes in governments, industries, companies, and/or our social life (Beheshti et al., 2016), that present frequent policy updates, a vast quantity of data, and complex data-flow. Business Processes (BPs) and their continuous improvements are central to the operation of companies. For those enterprises, business process analytics and verification constitute a key endeavor. Business Processes permit the description of the activities involved to achieve an objective in a company. The total or partial automation of processes creates an opportunity to gain insights into process execution and data analysis. It is crucial to analyse business processes and business process-related data captured in various information systems, that can be distributed in Big Data environments. Process data is stored across different systems, applications, and services, and is often shared between companies.

Organisations might use business processes and rules to describe their daily activities. Both aspects might evolve, due to changes in the laws or regulations, or to new behaviour within the companies. The modification of the control-flow model implies rebuilding the process and deploying it again, and therefore work-flow tends to become stable over time. However, the modification and updating of business rules, such as decision rules or policies, and the use of various types of data during the execution of various instances remains highly usual.

Fault diagnosis provides a mechanism to ensure the correct execution of the processes. In the business process context, there are certain specific changes derived from the nature of the business processes (Borrego & Gómez-López, 2019). One of the most relevant is the necessary efficiency of the fault detection process at run-time. Another characteristic is the strong relationship between the data and rules in various instances at the same time, where it is possible that the same data or rule can be shared by more than one instance or in various processes. This frequently occurs when data is stored in databases. These

* Corresponding author.
*E-mail addresses:* ceball@us.es (R. Ceballos), dianabn@us.es (D. Borrego), maytegomez@us.es (M.T. Gómez-López), gasca@us.es (R. M. Gasca).

https://doi.org/10.1016/j.eswa.2021.115212
Received 19 September 2019; Received in revised form 19 March 2021; Accepted 13 May 2021
Available online 28 May 2021

relations make the diagnosis process more complex, but they also assist in the isolation process, since a single diagnosis cannot contradict the correct behaviour of the whole process.

Since both data and rules can be modified at instantiation time, they cannot be included only in the design phase. Real contexts present highly modifiable environments, where both rules and data can evolve to be incorrect. Derived from the inclusion or modification of business rules and/or data, certain faults can be produced due to incorrect rules or data. In order to tackle this challenge, if an inconsistency is found, our approach is able to identify the origin of the problem. We propose a Hybrid Diagnosis methodology, where data used during the execution of the instances of the processes, and business rules, are simultaneously considered.

The relevance in analysing the data correctness is not a new issue in business processes (Gómez-López, Gasca, & Pérez-Álvarez, 2015), especially when relational databases are used. However, the extension to business rules has not been included before this work, nor the simultaneous combination of malfunction of business rules and data. The Business Compliance Rules (Becker, Ahrendt, Coners, Weiß, & Winkelmann, 2011) that can describe the data semantics and the relations between data values are named Business Data Constraints (henceforth referred to as BDCs) (Gómez-López et al., 2015; Gómez-López, Gasca, & Pérez-Álvarez, 2014). Business Compliance Rules can describe various types of behaviour in a company, such as the order of activities, agents who can execute the tasks, data value relation, etc. Business Data Constraints are a subset of Business Compliance Rules employed to describe the compliance relationship between the introduced data values and the business rules.

Two possible scenarios are presented in Fig. 1, Hybrid Diagnosis without using priorities, or Hybrid Diagnosis in accordance with the priorities between Data and Business Data Constraints. Business Data Constraints are designed by the Business Experts in accordance with the database. The minimal diagnosis is automatically obtained by solving a Constraint Optimization Problem. The priorities enable a ranking of the possible diagnoses, and therefore a more precise minimal diagnosis.

The methodology is divided into two parts that are shown in Fig. 1. The first part is dedicated to design the model at design time, which is performed by the Business Experts, and it includes Modelling the BDCs in accordance with the Database, and Selection of Criteria and Alternatives (based on an Analytic Hierarchy Process). The second part is dedicated to the automatic obtainment of the minimal diagnosis, which is executed at instantiation time. The two steps on the left side of Fig. 1, which are included in the Design of the Model, are human-based, and they are carried out by the business experts. The rest of the steps of Fig. 1 can be automatically executed. Our proposal tries to be user-friendly, and the expert is isolated of the process for obtaining the priorities and the generation of the Constraint Optimization Problem. However, the alternatives and the pairwise comparison of the alternatives are assessed by the expert.

The inclusion of business rules as a possible cause of malfunction brings about an exponential growth in the number of possible diagnosis. In previous work by Ceballos, Borrego, López, and Gasca (2016), the necessity to diagnose simultaneous combination of malfunction of data and rules in business process was detected. There is, however, a vast number of diagnoses that can be obtained when the database is very big and the rules can frequently change. An AHP provides a way to prioritise different possibilities of the cause of faults in the reasoning method, which is a necessary mechanism to reduce the number of possible diagnoses that can be found in complex problems such as that in this proposal. An AHP enables probabilistic constraints to be obtained according to the priority description regarding faults described by users and/or experts.

At instantiation time, the specific data from the data-flow and the database is analysed (Selection of the tuples of the DB related to the Business Data Constraints), and, depending on the data values and the instantiated BDCs, the priority relationships between data and rules are found, according to the likelihood of fault (Priorities of the alternatives). In this step, it is taken into account that not every item of data has the same probability of being incorrect (for example, long numerical data may be typed incorrectly more often than numbers of two digits). Our proposal incorporates priorities into the fault diagnosis method, to enable a ranking of probabilities of faults. There is a greater amount of data than of business rules and, in addition to this, the data is updated more frequently than are the rules. This is why it is more likely that a failure comes from an error in an item of data (in its insertion or update) than from an incorrect rule, although it remains possible to have faults in business rules. It is important to consider the probability of fault between data and business rules, and for this reason our methodology incorporates the likelihood concept, which enables the probabilities of faults in input data and business rules to be compared. Lastly, by means of Constraint Programming (Creation of the COP), the minimal diagnosis is obtained (Solving the COP for the automatic determination of the minimal diagnosis).

The paper is organised as follows. Section 2 presents an overview of related work found in the literature. Section 3 tackles the Model-based Diagnosis including data and business rules, isolating the possible minimal causes of non-conformance. Section 4 explains how the Constraint Programming paradigm can be employed to perform the diagnosis process. Section 5 presents how the priority models can help to reduce the complexity of the solution. Section 6 evaluates our proposal

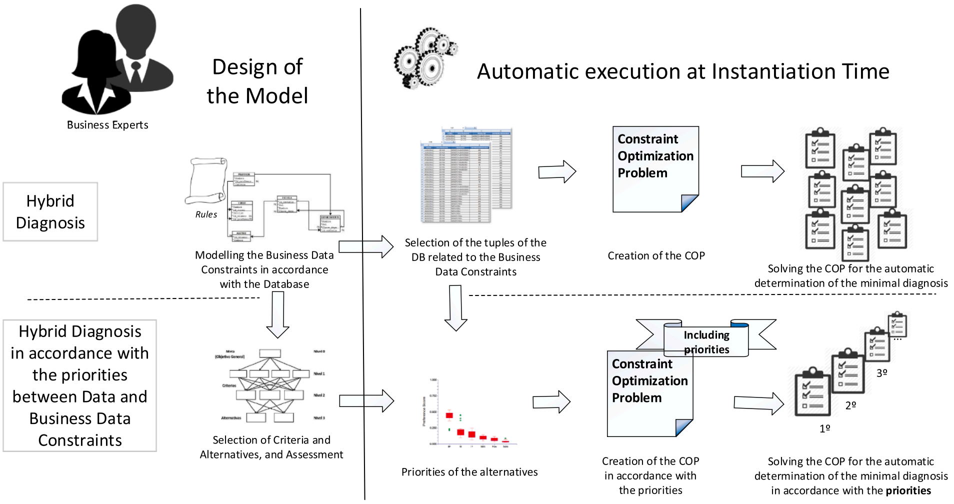

Business Experts
Design of the Model
Automatic execution at Instantiation Time
Hybrid Diagnosis
Rules
Modelling the Business Data Constraints in accordance with the Database
Selection of the tuples of the DB related to the Business Data Constraints
Constraint Optimization Problem
Creation of the COP
Solving the COP for the automatic determination of the minimal diagnosis
Hybrid Diagnosis in accordance with the priorities between Data and Business Data Constraints
Selection of Criteria and Alternatives, and Assessment
Priorities of the alternatives
Including priorities
Constraint Optimization Problem
Creation of the COP in accordance with the priorities
1º
2º
3º
Solving the COP for the automatic determination of the minimal diagnosis in accordance with the priorities

**Fig. 1.** Steps of the Methodology.

with a more complex example. In Section 7, the limitations of our proposal are enumerated. And finally, conclusions are drawn and future work is proposed in Section 8.

## 2. Related work

Papers related to data errors in Business Processes can be classified into two categories: data errors at design time; and data errors of BP instances at runtime.

First, and regarding the design-time analysis of the data model, one important aspect is the detection of the different types of errors that can occur in the data flow, such as missing, redundant, and conflicting data (Sun, Zhao, Nunamaker, & Sheng, 2006). This proposal has been extended to consider the analysis of business process models that take into account both the control flow perspective and the data flow perspective (Eshuis & Kumar, 2010; Sidorova, Stahl, & Trcka, 2011). Mechanisms to prevent errors at the structural level have been proposed (Trcka, van der Aalst, & Sidorova, 2009), but it is also necessary that the business processes comply with the established rules and policies. In Borrego, Eshuis, Gómez-López, and Gasca (2013, 2015), the semantics of activities are enriched and expressed by preconditions and postconditions that formally describe the behaviour that business process data should follow, and thus are used in design time to verify the correction of the model. In Pérez-Álvarez, López, Eshuis, Montali, and Gasca, 2020 the importance of the alignment between the workflow and the data storage structure is presented as a mandatory compliance analysis in process-aware information systems. The use of probability aspects in diagnosis is frequent to determine an order of possible fault explanations, even during the troubleshooting process for a better isolation, such as in Ramos et al., 2021. However, the comparison of different types of sources However, the comparison of different types of sources have not been considered.

In Maggi, Montali, and van der Aalst (2012) and Montali, Maggi, Chesani, Mello, and van der Aalst (2013), business constraint supervision is based on Event Calculus. Unlike the previous textual approaches, a visual language based on an extended Compliance Rule Graph (Knuplesch & Reichert, 2017) and a framework for visually monitoring all relevant perspectives of the compliance of business process (Knuplesch et al., 2017) have been proposed.

However, in enterprise business processes it is necessary to analyse the correctness of both rules and data, and to define compliance rules. In the literature, we find studies that perform this definition of data-aware compliance rules, such as Awad, Weidlich, and Weske (2011) and Ly, Rinderle-Ma, Knuplesch, and Dadam (2011), which define a notation that allows the relationships between compliance rules and data to be represented through data conditions. Likewise, Weidlich et al. (2011) propose a method to monitor the deviations that occur in control flow during the process execution. A deep analysis of the current state in business compliance patterns centred in data aspects is presented in Voglhofer and Rinderle-Mahofer, 2020, but unfortunately, an approach where both rules and data are the aim of the compliance is not analysed.

Moreover, previous studies in the literature address either the diagnosis of errors in the data of business processes (Gómez-López et al., 2014) or the diagnosis of errors in Business Data Constraints (BDCs) of business processes at runtime (Gómez-López et al., 2015). To the best of our knowledge, only in Ceballos et al. (2016) are both types of errors in business processes analysed at the same time. Our methodology is an extension of this previous work, but to the best of our knowledge, this proposal is the first that includes prioritization among data and rules, concretely it includes two main advantages: first, priorities are integrated to improve the precision of the identification of the causes of non-conformance between data and BDCs; and second, the computation of the weights for the goal function provides a more accurate process based on the probabilities of the possible distributions of data errors in BDC instances. Our approach based on priorities can reduce the number of possible diagnoses in complex scenarios where several rules and huge amounts of data are involved.

## 3. Conformance between data and business data constraints

When new business rules are designed, they must be in conformance with the stored data. However, there are several problems that must be solved before incorporating these new business rules. One of these problems is the verification of conformance between Data and BDCs. In this section, we introduce how to solve this problem and a real example.

### 3.1. The TCA example

In order to study the problem in hands, we introduce a financial application extracted from a real scenario in order to guide the explanation of the model and the steps of the methodology. The example is based on the activity of The Technological Corporation of Andalusia (TCA), which manages collaborative projects between private companies and research groups. The process manages the research project execution, which implies the research activities and expenditures executed to achieve the defined objectives. Numerous members of the personnel modify the information related to the execution of more than 300 projects. Data is stored in a relational database formed of 86 tables. Project data is entered by employees, with an average of two hundred items of data per employee and project for, at most, the 3 or 4 years of each project. Laws or regulations applied to the projects have changed in recent years and therefore business rules can be modified, which can involve a change in their compliance with the former data.

In order to comply with laws and requirements applied to the projects, the idea involves designing and adding/modifying business rules to the business processes. These new business rules must be in conformance with the stored data. However, there are several problems that must be solved before incorporating these new business rules:

- Certain values of the database can contain errors due to human mistakes. These errors must be modified in order to store only correct information, which satisfies the business rules.
- Certain business rules can be badly designed, and therefore the values stored in the database would not satisfy these rules. This makes their redefinition necessary, so that only well-designed rules are added.
- Certain laws and requirements applied to the projects have changed in recent years, and therefore certain business rules can only be applied to business processes or to data stored associated with a range of years.

The goal of this paper is to obtain an automatic methodology for the isolation of mistakes in stored data and business rules before the business rules are added to the business process.

### 3.2. Description of the business data constraint

During the execution of a business process, the data flowing through the process can be read and updated. This data is usually obtained and stored in a relational database, in order to maintain its persistence. It is possible that the same data is being read and written in other instances of other processes. Therefore, data (stored in databases) constitutes a fundamental component in Information System and Business Process Management (van der Aalst, ter Hofstede, & Weske, 2003).

To ensure that the values of the data used during the process instances are compliant with the policies of the business model, Business Data Constraints are used.

**Definition 1.** (*Business Data Constraint (BDC)*) Rule that represents the semantic relationships that exist between the data that is entered, accessed, and updated during the execution of the different business process instances. BDCs describe the correct values of relationships between the data involved.

BDCs are a type of business compliance rules. In this paper, we suppose that BDCs are presented as Numerical Constraints following the grammar shown in Fig. 2. It is important to note that the variables involved in the constraints can store values from both the system database and the execution data flow. The following BDCs are a subset of the policies of the previous example (Section 3.1). These BDCs represent data relations and are associated with various activities.

- $\mathrm{BDC}_1$: hardwareCost + softwareCost + humanCost = totalCost
- $\mathrm{BDC}_2$: subsidisedCost $\leqslant$ totalCost
- $\mathrm{BDC}_3$: subsidisedPerYear $\leqslant$ potentialSubsidised
- $\mathrm{BDC}_4$: humanCostPerYear $\leqslant$ maximumHuman
- $\mathrm{BDC}_5$: subsidisedCost $\geqslant 3 \cdot$ subsidisedPerYear

The variables that participate in the BDCs correspond to the values stored in the database, which were instantiated during execution. A part of the relational model referring to the previous example, composed of three tables, is shown in Fig. 3. This model stores the data of the projects (*Project*), their details per year (*ProjectPerYear*), and the spending limits allowed for each year and for each item (*LimitsPerYear*).

### 3.3. Selection of the Tuples of the Relational Database

Relational databases tend to be used as a mechanism to maintain data generated during process execution. It is possible for a simple BDC to relate attributes from several tables in the database. The fact that the data is located in different tables is due to the need to follow the *Normal Forms* principle defined in the theory of relational databases. Thanks to the normalization rules, possible update anomalies and inconsistencies between the items of data can be prevented. If these items of data are involved in the same BDC, it means they are related despite the fact that the data could be stored in different tables.

The storage of data in different tables makes it difficult to verify compliance with the BDCs. Therefore, a *denormalisation process* is carried out that results in a new structuring of the data, but which will only be used in the conformance process, meaning that no changes are made in the relational database.

The denormalisation process obtains a join-table where all attributes are together. The join-table includes all the attributes of the tables of Fig. 3. The join-table is shown in Fig. 5, which is obtained from the values of Fig. 4 and takes into consideration the relational model of Fig. 3. The attributes *idProject* from *Project* and *ProjectPerYear* tables are related through a primary-foreign key relationship. And similarly, for the relationship between tables *LimitsPerYear* and *ProjectPerYear*.

All the variables that appear in a BDC have a related attribute in the join-table. It should be taken into account that, after this denormalisation process, the same value of an attribute may appear in different tuples of the new join-table, due to the existing $1..n$ relationships between certain tables. A new column is then included for each attribute during the denormalisation process, whose purpose is to name the different valuations in the database. The word Id is added to the name of the attribute, and the column stores a different identification (integer number) for each value of the related attribute. These identifications are necessary for it to be discerned whether two equal values in the join-table come from a single value before commencing the denormalization process, since the appearance of two equal values in the same column does not imply necessarily that they come from a single value in the normalised database (and vice versa).

For example, the value associated with the human cost of project 121 is 45000, and this appears in the first two tuples of Fig. 5, while the column *humanCostId* stores the same identification in the first two tuples since this project has two years of activity; this is the same value that appears in the first tuple of the table *Project* (Fig. 4). As another example, the human cost per year of project 121 and year 2015 is 22,500, and it is the same as the human cost per year of project 121 and year 2016, but the column *humanCostPerYearId* stores a different value for each case (1 and 2) because they correspond to two different tuples of the table *ProjectPerYear* before the denormalisation process.

### 3.4. Verification of conformance between Data and BDCs

For the verification of the conformance of the BDCs and the data, the BDCs are instantiated using the newly obtained tuples. For the example, its BDCs are instantiated with the tuples shown in Fig. 5. Several of the instances are given in Table 1. In Table 2, a summary of the compliance with the BDCs is presented. For each $BDC_j^i$, $j$ is the index of the BDC, and $i$ is the index of the instance. For example, $\mathrm{BDC}_1$ (softwareCost + hardwareCost + humanCost = totalCost) has two different types of tuples ({1, 2} and {3, 4, 5}), and hence two different BDCs are created in accordance with the tuples $BDC_1^1$ and $BDC_1^2$. $BDC_4$ is applied over five different tuples, thereby creating five different BDCs in accordance with the tuples $BDC_2^1$, $BDC_2^2$, $BDC_2^3$, $BDC_2^4$ and $BDC_2^5$.

Subsequent to the instantiation of the BDCs with the obtained tuples, $BDC_3^5$, $BDC_4^1$, $BDC_4^3$, $BDC_5^1$, $BDC_5^2$, $BDC_5^3$, and $BDC_5^4$ become unsatisfiable. The objective is to ascertain, in an automatic way, which minimal set of BDCs and values of the database must be changed in order to clarify all this non-conformance behaviour.

### 3.5. The possible minimal set of incorrect input values and BDCs

Our approach is based on concepts and definitions of the Model-based Diagnosis Methodology (MBD) which, in turn, is based on a model named System Description (SD) and a set of values named Observations (OBS). In our case, the elements of SD are the set of BDCs, and the OBSs include the values stored in the database. If the instances of the BDCs are not satisfied when using the values stored in the database, then this non-conformance behaviour can have two kinds of explanation:

- The design of one or more BDCs contains defects. The solution is to change these badly designed BDCs. Notice that if a BDC is incorrect, certain tuples where this BDC is applied can be unsatisfiable, while other tuples can be satisfied, although only one BDC would be indicated as responsible for the malfunction.

```
BusinessDataConstraint := Atomic_Constraint BOOL_OP BusinessDataConstraint
  | Atomic_Constraint | 'NOT' Constraint
BOOL_OP := 'AND' | 'OR' | '→'
Atomic_Constraint := function PREDICATE function
function := Variable FUNCTION_SYMBOL function
  | Variable | Constant
Variable := Table'.'Attribute | Attribute | DataFlowVariable | Constant
PREDICATE := '=' | '<' | '≤' | '>' | '≥'
FUNCTION_SYMBOL := '+' | '−' | '*' | '/'
```

**Fig. 2.** Grammar Model.

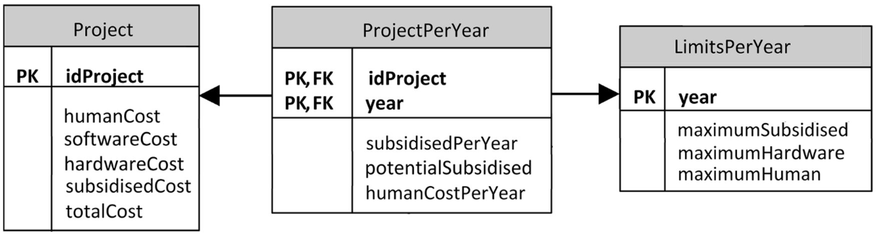


**Fig. 3.** Relational Model.

| Project | | | | | |
|---|---|---|---|---|---|
| idProject | subsidisedCost | softwareCost | hardwareCost | humanCost | totalCost |
| 121 | 55000 | 11000 | 24000 | 45000 | 80000 |
| 122 | 58000 | 6000 | 20000 | 36000 | 62000 |
| … | … | … | … | … | |

| ProjectPerYear | | | | |
|---|---|---|---|---|
| idProject | year | subsidisedPerYear | potentialSubsidised | humanCostPerYear |
| 121 | 2015 | 27500 | 28500 | 22500 |
| 121 | 2016 | 27500 | 27500 | 22500 |
| 122 | 2015 | 22000 | 24000 | 14000 |
| 122 | 2016 | 21000 | 22000 | 12000 |
| 122 | 2017 | 25000 | 22000 | 10000 |
| … | … | … | … | … |

| LimitsPerYear | | | |
|---|---|---|---|
| year | maximumSubsidised | maximumHardware | maximumHuman |
| 2015 | 28000 | 12000 | 13000 |
| 2016 | 30000 | 12500 | 24000 |
| 2017 | 22000 | 10000 | 25000 |
| … | … | … | … |

**Fig. 4.** Example of tuples for the Relational Model of Fig. 3.

| BDCs: | | (2,5) | | (1) | | (1) | | (1) | | (1,2) | | | (3,5) | | (3) | | (4) | | | | | | (4) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| tuple | idProject | subsidisedCostId | subsidisedCost | softwareCostId | softwareCost | hardwareCostId | hardwareCost | humanCostId | humanCost | totalCostId | totalCost | year | subsidisedPerYearId | subsidisedPerYear | potentialSubsidisedId | potentialSubsidised | humanCostPerYearId | humanCostPerYear | maximumSubsidisedId | maximumSubsidised | maximumHardwareId | maximumHardware | maximumHumanId | maximumHuman |
| 1 | 121 | 1 | 55000 | 1 | 11000 | 1 | 24000 | 1 | 45000 | 1 | 80000 | 2015 | 1 | 27500 | 1 | 28500 | 1 | 22500 | 1 | 28000 | 1 | 12000 | 1 | 13000 |
| 2 | 121 | 1 | 55000 | 1 | 11000 | 1 | 24000 | 1 | 45000 | 1 | 80000 | 2016 | 2 | 27500 | 2 | 27500 | 2 | 22500 | 2 | 30000 | 2 | 12500 | 2 | 24000 |
| 3 | 122 | 2 | 58000 | 2 | 6000 | 2 | 20000 | 2 | 36000 | 2 | 62000 | 2015 | 3 | 22000 | 3 | 24000 | 3 | 14000 | 1 | 28000 | 1 | 12000 | 1 | 13000 |
| 4 | 122 | 2 | 58000 | 2 | 6000 | 2 | 20000 | 2 | 36000 | 2 | 62000 | 2016 | 4 | 21000 | 4 | 22000 | 4 | 12000 | 2 | 30000 | 2 | 12500 | 2 | 24000 |
| 5 | 122 | 2 | 58000 | 2 | 6000 | 2 | 20000 | 2 | 36000 | 2 | 62000 | 2017 | 5 | 25000 | 5 | 22000 | 5 | 10000 | 3 | 22000 | 3 | 10000 | 3 | 25000 |
| .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. | .. |

**Fig. 5.** Join-table following the denormalisation process.

- One or more values stored in the database are incorrect.

The goal of our methodology is to obtain a minimal set of elements that is able to explain the non-conformance behaviour. This minimal set is named minimal diagnosis. It is based on the parsimony principle (Peng & Reggia, 1990), which determines the simplest explanation to describe a beheaviour, that when applied to model-based diagnosis implies selecting the minimum set of faults.

**Definition 2.** (*Minimal diagnosis*)  is a subset $MD \subseteq (SD \cup OM)$, in such a way that if $SD$ is not satisfied, then $SD - MD$ can be satisfied, and for all

**Table 1**
Conformance between Data and BDCs

| BDC | Tuple | $BDC_i^j$ | Right |
|---|---|---|---|
| 1 | 1,2 | $BDC_1^1$ | softwareCost1 + hardwareCost1 + humanCost1 = totalCost1<br>11000 + 24000 + 45000 = 80000 ✓ |
| 1 | 3,4,5 | $BDC_1^2$ | softwareCost2 + hardwareCost2 + humanCost2 = totalCost2<br>6000 + 20000 + 36000 = 62000 ✓ |
| 2 | 1,2 | $BDC_2^1$ | subsidisedCost1 ⩽totalCost1<br>55000 = 80000 ✓ |
| 2 | 3,4,5 | $BDC_2^2$ | subsidisedCost2 ⩽totalCost2<br>58000 = 62000 ✓ |
| … | … | … | … |
| 4 | 1 | $BDC_4^1$ | humanCostPerYear1 ⩽maximumHuman1<br>23000 ⩽13000 × |
| 4 | 2 | $BDC_4^2$ | humanCostPerYear2 ⩽maximumHuman2<br>22000 ⩽24000 ✓ |
| 4 | 3 | $BDC_4^3$ | humanCostPerYear3 ⩽maximumHuman1<br>14000 ⩽13000 × |
| 4 | 4 | $BDC_4^4$ | humanCostPerYear4 ⩽maximumHuman2<br>12000 ⩽24000 ✓ |
| 4 | 5 | $BDC_4^5$ | humanCostPerYear5 ⩽maximumHuman3<br>10000 ⩽25000 ✓ |
| … | … | … | … |

**Table 2**
Evaluation of the compliance with the BDCs.

| BDC | Tuple | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| 1 | | $BDC_1^1$ ✓ | | $BDC_1^2$ ✓ | |
| 2 | | $BDC_2^1$ ✓ | | $BDC_2^2$ ✓ | |
| 3 | $BDC_3^1$ ✓ | $BDC_3^2$ ✓ | $BDC_3^3$ ✓ | $BDC_3^4$ ✓ | $BDC_3^5$ × |
| 4 | $BDC_4^1$ × | $BDC_4^2$ ✓ | $BDC_4^3$ × | $BDC_4^4$ ✓ | $BDC_4^5$ ✓ |
| 5 | $BDC_5^1$ × | $BDC_5^2$ × | $BDC_5^3$ × | $BDC_5^4$ × | $BDC_5^5$ ✓ |

$D \subset MD : (SD \cup OM) - D$ cannot be satisfied.

A minimal diagnosis is the minimal set of BDCs and values of the database that must be changed in order to be able to satisfy all the instances of the BDCs. In Section 4, the implementation of our methodology is explained, together with how to obtain the minimal diagnosis by transforming the knowledge of the system, regarding the diagnosis of the values stored in the database and the BDCs, into constraints. There can be more than one minimal diagnosis that is a solution to the diagnosis problem. For these cases, in Section 5, the specifications of different types of priorities are included in order to improve the precision of the diagnosis methodology and to reduce the number of possible minimal diagnoses.

The isolation of the incorrect values entails a certain degree of complexity, derived from the number of values that are affected by a BDC. Whenever more than one value can provide the explanation for a malfunction, there may be a simpler explanation based on making only one BDC responsible for it, since a single BDC may involve all possible incorrect values. Hence, this single BDC can be considered as the only component required for the explanation of the malfunction. However, since data is more frequently introduced in the process, in an intuitive way it is therefore more probable to obtain an incorrect value than an incorrect BDC. In order to formalise this intuitive thinking, a definition of the weight to balance this relation should be included.

**Definition 3**. (*Probability between values and BDC malfunction*) The percentage that describes the probability of the appearance of a fault in the data of a BDC. For example, a probability of 60% means that a BDC will be designated as responsible for a fault only if more than 60% of the instances (tuples of the joined table created for the variables involved in this BDC) are unsatisfiable for that BDC.

The tuples introduced into the database can have different origins (i.e., input variables, triggered in the database, or derived in the activity) which will affect the diagnosis process. The main difference lies between the introduced and derived values. Input values are introduced by the user and can be typed incorrectly. However, derived values are calculated/created by using other values. This implies that derived values cannot be responsible for a malfunction, but they can be related to incorrect input values.

## 4. Creation of constraint optimization problem for model-based diagnosis

The verification of the correction of a system can be performed by means of the analysis of the satisfiability of the model $\{SD, OM\}$, which represents the conformity of the OM and the BDCs that describe the SD. When a non-satisfiability is detected, then the next issue is to determine what is responsible for the malfunction: the minimal explanation following the parsimony principle that can be either BDCs (SD) or the input values (OM). Formally, a minimal diagnosis must be found. Derived from the similarity of the BDCs and the arithmetic constraints, we propose the use of the Constraint Programming paradigm (Rossi, van Beek, & Walsh, 2006) to infer the minimum set of constraints that describe the malfunction. This section describes how to create a Constraint Satisfaction Problem (CSP) for the subsequent automatic determination of the minimum explanation of a malfunction that combines *SD* and *OM*.

The Constraint Programming paradigm includes a set of algorithms applied to Constraint Satisfaction Problems (CSP) in order to determine the values of a set of variables in a domain that satisfy a set of constraints. Formally, a CSP is described by the tuple $\langle V, D, C\rangle$, to describe the variables, the domains, and the constraints, respectively. The assignment of the values of the variables must satisfy the constraints. The resolution of a CSP implies discovering, in an efficient way, the values of the variables that satisfy the constraints. The possible values of the variables obtained can be very wide-ranging, since they depend on the domain, whereby it is possible to obtain the first solution found, every solution, or a specific solution among those that minimise or maximise one of the variables of the problem. In this last case, the CSP is called a Constraint Optimisation Problem (COP), where the goal is to minimise (Min-CSP) or maximise (Max-CSP) an optimisation function ($f$). Since the objective of model-based diagnosis is to minimise the possible explanation, the Min-CSP will be used herein.

### 4.1. *Constraints-based model for diagnosis*

The first step in obtaining the diagnosis entails the translation of the problem into a CSP, including BDCs and tuples of the join-table. To tackle the problem, the BDCs associated with the whole process or associated with each activity of the process must be combined according to the control-flow of the process. In previous work, Gómez-López et al. (2014) analyse how to combine the BDCs in a single CSP. Their proposed algorithm traverses the business process model and combines the BDCs related to each activity.

The Min-CSP obtained by transforming the example is shown in Fig. 6. Below, the process for modelling the Min-CSP is given in detail. A Min-CSP includes variables, constraints, and an objective function.

- **Variables of the problem**. All the variables that appear in a BDC have a related attribute in the join-table. For each of these attributes, a set of $m$ variables is added to the CSP. For example, in the join-table (Fig. 5), the column *maximumHuman* stores the values while the column *maximumHumanId* stores the identification of the values

**Variables of the problem:**

integer maximumHuman1, maximumHuman2, maximumHuman3, humanCostPerYear1, ...
integer[0,1] $\mathrm{rMh}_1, \mathrm{rMh}_2, \mathrm{rMh}_3, \mathrm{rHpy}_1, \mathrm{rHpy}_2, \mathrm{rHpy}_3, \mathrm{rHpy}_4, \mathrm{rHpy}_5, \ldots$
$\ldots, \mathrm{rBDC}_4, \mathrm{rBDC}_4^1, \mathrm{rBDC}_4^2, \mathrm{rBDC}_4^3, \mathrm{rBDC}_4^4, \mathrm{rBDC}_4^5, \mathrm{rBDC}_5, \mathrm{rBDC}_5^1, \ldots, \mathrm{rBDC}_5^5$

**Constraints related to the instances of variables:**

$\mathrm{rMh1} = \neg(\mathrm{maximumHuman1} = 13000) \qquad \mathrm{rMh2} = \neg(\mathrm{maximumHuman2} = 24000)$

$\ldots$

$\mathrm{rHpy1} = \neg(\mathrm{humanCostPerYear1} = 23000) \qquad \mathrm{rHpy2} = \neg(\mathrm{humanCostPerYear2} = 22000)$

$\ldots$

**Constraints related to the instances of BCDs:**

$\ldots$

$$\mathrm{rBDC}_4^1 = \neg(\mathrm{humanCostPerYear1} \leq \mathrm{maximumHuman1})$$
$$\mathrm{rBDC}_4^2 = \neg(\mathrm{humanCostPerYear2} \leq \mathrm{maximumHuman2})$$
$$\mathrm{rBDC}_4^3 = \neg(\mathrm{humanCostPerYear3} \leq \mathrm{maximumHuman1})$$
$$\mathrm{rBDC}_4^4 = \neg(\mathrm{humanCostPerYear4} \leq \mathrm{maximumHuman2})$$
$$\mathrm{rBDC}_4^5 = \neg(\mathrm{humanCostPerYear5} \leq \mathrm{maximumHuman3})$$

$\ldots$

$$\mathrm{rBDC}_4 = (\mathrm{rBDC}_4^1 + \mathrm{rBDC}_4^2 + \mathrm{rBDC}_4^3 + \mathrm{rBDC}_4^4 + \mathrm{rBDC}_4^5 \geq \mathrm{minLik}_4)$$

$\ldots$

$$(\mathrm{rBDC}_4^1 + \mathrm{rBDC}_4^2 + \mathrm{rBDC}_4^3 + \mathrm{rBDC}_4^4 + \mathrm{rBDC}_4^5 = 0) \vee$$
$$(\mathrm{rBDC}_4^1 + \mathrm{rBDC}_4^2 + \mathrm{rBDC}_4^3 + \mathrm{rBDC}_4^4 + \mathrm{rBDC}_4^5 \geq \mathrm{minLik}_4)$$

$\ldots$

**Objective function:**

$\mathrm{minimize}(\mathrm{rMh}_1 + \mathrm{rMh}_2 + \mathrm{rMh}_3 + \mathrm{rHpy}_1 + \mathrm{rHpy}_2 + \mathrm{rHpy}_3 + \mathrm{rHpy}_4 + \mathrm{rHpy}_5 + \ldots$
$\ldots + \mathrm{rBDC}_4 \cdot \mathrm{minLik}_4 + \mathrm{rBDC}_5 \cdot \mathrm{minLik}_5)$

**Fig. 6.** Min-CSP example.

stored in column *maximumHuman*. For the attribute *maximumHuman*, a total of *m* variables are added to the Min-CSP, where *m* is 3 since this it is the total number of different identifications stored in the column *maximumHumanId*. These 3 variables are named by add-ing the identification to the name of the attribute: *maximumHuman*1, *maximumHuman*2 and *maximumHuman*3 (Fig. 6). In brief, for each instance of an input variable *k*, a variable is defined:

$$type\ var_k^1, \ldots, var_k^m, \ldots$$

To isolate the type of source that produces the malfunction (i.e., data and/or BDCs), CSPs are created by using reified constraints that assign a truth value to a constraint to ascertain whether it can be satisfiable or not. These variables are associated to each $BDC_i$ or $\mathrm{BDC}_i^j$ (instance j of $BDC_i$) and to each assignment of a value to each $\mathrm{var}_k^m$ (instance m of the input variable $\mathrm{var}_k$), in order to ascertain the satisfiability:

$$integer[0,1] \ldots, rVar_k^m, \ldots, rBDC_i, \ldots, r\mathrm{BDC}_i^j, \ldots$$

The domain of these variables only includes value zero (false value) or value one (true value).

- **Constraints related to the instances of variables**. For each instance *m* of an input variable *k*, a new constraint is added to the Min-CSP:

$$rVar_k^m = \neg\left(var_k^m = value_k^m\right)$$

The reified variable $\mathrm{rVar}_k^m$ is equalised to the negated constraint since the goal is to obtain the minimal number of elements with abnormal behaviour. For example, in constraint $rMh1 = \neg(maximumHuman1 = 13000)$, if *rMh*1 is 1, then it is supposed that the value 13000 is erroneous (abnormal behaviour), and it must be changed, otherwise if *rMh*1 is 0, then it is supposed that the value 13000 is correct (normal behaviour).

- **Constraints related to the instances of BCDs**. For each instance $\mathrm{BDC}_j^i$, a new constraint is added in order to represent the satisfiability:

$$rBDC_i^1 = \neg\left(BCD_i\ instantiated\ by\ tuple\ 1\right)$$
$$\ldots$$
$$rBDC_i^n = \neg\left(\mathrm{BDC}_i\ instantiated\ by\ tuple\ n\right)$$

$rBDC_i^j$ represents the satisfiability of each instance $\mathrm{BDC}_j^i$. For example, in constraint $rBDC_4^1 = \neg\,(humanCostPerYear1 \leqslant maximumHuman1)$, if $rBDC_4^1$ is 0, then $humanCostPerYear1 \leqslant maximumHuman1$ (normal behaviour), and if $rBDC_4^1$ is 1, then $humanCostPerYear1 > maximumHuman1$ (abnormal behaviour). In order to represent in the CSP when a malfunction in an item of data is more likely than in a BDC, it is necessary to include the following constraint for each BDC:

$$rBDC_i = rBDC_i^1 + \ldots + rBDC_i^n = \sum_j^n rBDC_i^j \geqslant minLik_i$$

Both $\mathrm{rBDC}^i$ and $\mathrm{rBDC}_i^j$ are necessary to distinguish the general BDC from the constraint obtained when it is used in each tuple. This provides a way to incorporate the likelihood into the problem, through the parameter $\mathrm{minLik_i}$. The parameter $\mathrm{minLik_i}$ is a threshold which is equal to the minimum number of non-compliant instances of a $BDC_i$, which determines that the problem is in the design of $BDC_i$ and not in the input values of the instances. For example, if there are ten instances of $BDC_i$ then $minLik_i$ can take a value between 1 and 10. If the number of instances that are not satisfied is equal to or greater than the $minLik_i$ threshold, then $BDC_i$ becomes a part of the minimal diagnosis. In SubSection 4.2, it is shown how the value of each $minLik_i$ is calculated.

Finally, it is necessary to add the following constraint for each $BDC_i$:

$$(rBDC_i^1 + \ldots + rBDC_i^n = 0) \vee (rBDC_i^1 + \ldots + rBDC_i^n \geqslant minLik_i)$$

There are two options: (1) $\sum_j^n rBDC_i^j$ is equal to zero, and therefore the $BDC_i$ is correct; or (2) $\sum_j^n rBDC_i^j$ is equal to or greater than minLik$_i$, and therefore the $BDC_i$ has a defect. Values between one and minLik$_i$-1 are not allowed. In other words, if $\sum_j^n rBDC_i^j$ is zero and any single instance $BDC_i^j$ is not satisfied by using the values stored in the data base, then there must be a error in one of these values, but $BDC_i$ is correct.

- **Objective function**. This is defined as:

$$minimize\left(rVar_k^1 + \ldots + rVar_k^m + \ldots + rBDC_1 \cdot minLik_1 + \ldots + rBDC_q \cdot minLik_q\right)$$

The objective is a weighted function where each ***rBDC**$_i$* is associated with a weight equal to the parameter minLik$_i$. To minimise the goal function and to obtain the minimal diagnosis, the solver will seek to assign the value 0 to the greatest number of variables included in the goal function.

### 4.2. Computation of the MinLik parameter

The *minLik* parameter depends on several factors. The computation for the proposed example is shown in Table 3, and the explanation of the columns is the following:

- *nInst*: number of instances of $BDC_i$.
- *nVar*: number of variables involved in all instances (number of different types of tuples) of $BDC_i$. For example, for BDC$_1$, there are two different types of tuples ({1, 2, 3} and {4, 5}) as shown in Section 3.3, and *nVar* is 8 because this is the total number of different variables in these two instances. For $BDC_4$, there are five instances, and *nVar* is 8.
- *%errors*: average percentage of data errors. This is estimation provided by an expert, and it should be based on statistics obtained on previous data errors.
- *nErrors*: likely number of data errors of $BDC_i$. The value of this variable is derived from the rounded nearest Integer of *nVar* · *%errors*.
- *range*: minimal and maximal number of instances of a BDC that can contain data errors. It represents the interval of instances that can contain errors if the total number of data errors is equal to *nErrors*. For example, there are 5 instances of $BDC_4$ with a total of 8 input variables and *%Errors* is 20%, then *nErrors* is $2 \approx 1.6 = 8 \cdot 0.2$. For $BDC_4$ the *range* is [1, 4]. $BDC_4$ contains 2 input variables for each instance, the minimal value of the *range* is 2 because there can be cases where only 1 instance contains the 2 data errors. The maximal number of the *range* is 4, which is the case when there is a data error in variable *maximumHuman*1 (tuples 1 and 3) and there is a data error in variable *maximumHuman*2 (tuples 2 and 4).

  Fig. 7 shows the probability for each possible distribution of data errors in BDC instances, in an example with 20 instances and *%Errors* equal to 20%. In this example, the instances of the BDC contain a total of 80 input variables, and therefore *nErrors* is $16 = 80 \cdot 0.2$. If all these variables are different, then the *range* is [4, 16]. The maximal number of instances that should contain a data error is 16, which is the case when there is one data error for each instance. If this BDC contains four variables for each instance, then the minimal is 4 because that is the case when four instances contain four data errors. The probability for each possible distribution is calculated by taking into account all possible cases, and it is supposed that all cases are equiprobable. As a consequence, if the BDC has been instantiated with the values of the obtained tuples, and if there are faults in more than 16 instances, then the problem is probably due to a defect in the definition of the BDC, since 16 is the maximal number of affected instances if *range* = [4, 16] (a random distribution of data errors is assumed). In Fig. 7, the accumulated probability is also shown; for example, the accumulated probability for the interval [4, 12] is a nearly 70%, and for [4,13] this exceeds 90%.
- *minLik*: the calculation is performed by adding 1 to the first element of the *range* where the accumulated probability is greater than 85%. For example, in Fig. 7, the value 13 of the interval has an accumulated probability equal to 90%, and the value for the *minLik* parameter should be 1 + 13 = 14.

**Table 3**
Obtained values of *minLik* parameter.

| BDC | nInst | nVar | %errors | nErrors | range | minLik |
|---|---|---|---|---|---|---|
| 1 | 2 | 8 | 20% | $1.6 \approx 2$ | [1, 2] | 2 |
| 2 | 2 | 4 | 20% | $0.8 \approx 1$ | [1, 1] | 2 |
| 3 | 5 | 10 | 20% | $2 \approx 2$ | [1, 2] | 3 |
| 4 | 5 | 8 | 20% | $1.6 \approx 2$ | [1, 4] | 4 |
| 5 | 5 | 7 | 20% | $1.4 \approx 1$ | [1, 3] | 3 |

### 4.3. Determining the minimal diagnosis

The solution of the Min-CSP problem is given by the minimal diagnosis. The minimal diagnosis for the example presented in Fig. 6 is a set of three elements that must be modified:

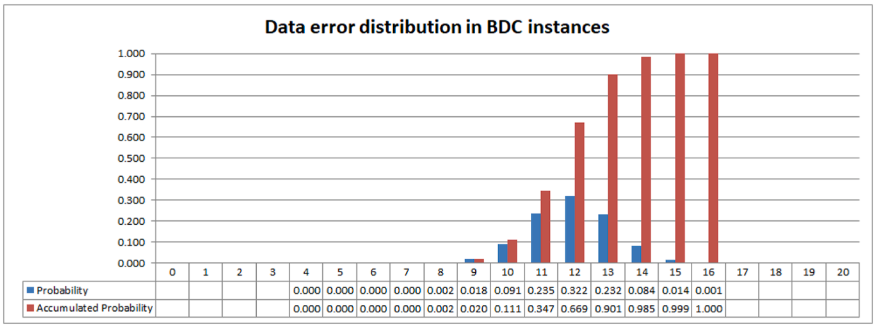
- The value associated with the variable *maximumHuman*1 (used in some instances of *BDC*$_4$) is erroneous; the correct value is 23,000.
- The *BDC*$_5$ is not correct, the correct BDC should be the expression *subsidisedCost* $\geqslant 2 \cdot$ *subsidisedPerYear*.
- Finally, the value associated with *subsidisedPerYear*5 or with *potencialSubsidised*5 (both were used in one instance of *BDC*$_3$) is erroneous. In this case, the correct value for *subsidisedPerYear*5 is 15,000.

If and only if these changes are made, can all the BDCs be satisfied by applying the stored values.

## 5. Priorities model-based diagnosis using Analytic Hierarchy Process

The incorporation of the parameter *minLik* helps to distinguish the likelihood between data and BDCs in general, but usually not every item of data and BDC has the same likelihood of occurring, due to factors such as the person who introduces them, the size of the data or BDCs, and whether they have already been stored for a long time in the system. In order to improve the precision of the fault diagnosis, it is necessary to include a Decision Support System (DSS) that facilitates the specification of different types of priorities to data, BDCs, or the combination of both. For this reason, our proposal involves the Analytic Hierarchy Process (AHP) (Saaty, 2008), which permits the assignation of priorities so that not all of the data and BDCs receive the same consideration.

The business analysts and a team of diagnosis experts are involved throughout the whole process of the DSS in establishing the list of possible fault priorities, whereas the business process managers are involved in setting up response plans. Our methodology helps in the description of these priorities and incorporates them automatically into the Min-CSP. It brings out the reduction of the minimal diagnosis, by offering the most likely diagnosis first according to the described priority.



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 🟦 Probability | | | | | 0.000 | 0.000 | 0.000 | 0.000 | 0.002 | 0.018 | 0.091 | 0.235 | 0.322 | 0.232 | 0.084 | 0.014 | 0.001 | | | | |
| 🟥 Accumulated Probability | | | | | 0.000 | 0.000 | 0.000 | 0.000 | 0.002 | 0.020 | 0.111 | 0.347 | 0.669 | 0.901 | 0.985 | 0.999 | 1.000 | | | | |

**Fig. 7.** Probability for each possible distribution of data errors in BDC instances.

### 5.1. *Analytic hierarchy process*

When multiple instances of a business process are executed, information is gathered to help us understand said process in a significant way. In order to make better decisions regarding the possible faults in the multiple instances, these decisions will not be made by intuitively measuring intangible factors, but instead all kinds of information of the instances are considered to be useful, and the greater the quantity, the better. These goals indicate the selection of the AHP as a better method for their compliance.

The existence of multiple alternatives to be chosen depending on multiple aspects or criteria has been included in several scenarios (Köksalan, Mousseau, Ozpeynirci, & Bilgin Ozpeynirci, 2009; Pereira, Figueira, Mousseau, & Roy, 2009; Zyoud & Fuchs-Hanuschoud, 2017). We propose the use of AHP for the prioritisation the possible faults. It enables the creation of priorities by the criteria themselves, in order to weigh the priorities of the possible faults and add the criteria to obtain the desired ranking of the faults. Multicriteria decision analysis (MCDA) deals with these problems and AHP is often employed to make decisions regarding the generation of these priorities. According to Saaty (2008) and Saaty (2000), the decision process is decomposed into the following steps: Goal Definition Phase, Selection Phase of Criteria and Alternatives, Assessment Phase, and Prioritisation Phase. In subsequent paragraphs, the application of the decision process according to the previous steps is presented.

### 5.2. *Goal definition phase*

In this phase, the problem is defined and the goal of the decision is specified. The goal is to diagnose faulty data or BDCs in multiple instances of a business process. For each table of a database, an attribute is a vertical column which contains a set of values. The possible faults can be defined for attributes, for BDCs, or for BDCs and attributes.

### 5.3. *Selection phase of criteria and alternatives*

In this phase, the decision hierarchy is established from the top with the goal of the decision, through the intermediate levels (criteria), to the lowest level (alternatives). The criteria are dependent on the context of each particular problem and are hence defined by the business expert. The criteria to compare alternatives should not change during the execution of the business process. The definition of the criteria aims to compare BDCs and/or data in order to establish their hypothetical ranking as being responsible for a fault in runtime.

Depending on the case of study, there can be different types of criteria: criteria to differentiate between items of data, criteria to differentiate between BDCs, or criteria for any pairwise comparison between data and BDCs. In particular, and for the business processes in the context of this paper, we suggest several examples of criteria in order to reflect the idea:

- Fault probability: this criterion enables a comparison to be made between data and BDCs, and indicates the probability of a particular cluster of elements being responsible for a fault against any other cluster of elements.
- Human reliability: this enables a comparison to be made of the data, based on the reliability of the human in charge of the inclusion of each value in the database, and defined by means of human roles.
- Source reliability: this enables a comparison to be made between the trustworthiness of the source of different clusters of data and/or BDCs.
- Data accuracy: the aim of this criterion is to determine the required accuracy of each cluster of data. That is, the larger an item of data is, the easier it is to make a typo (for example, a bank account is more liable to be incorrect than a person's age), and the possible influence of the data in the appearance of errors is even considered (again, an incorrect bank account is more likely to cause faults than an incorrect age).
- BDC persistence: the longer a cluster of BDCs had remained in the model (that is, the older it is), the more probable it is that those BDCs have already been tested and verified.

The single alternatives could be: $BDC_1 \ldots BDC_n$, $BDC_m \ldots BDC_p$, $attribute_1 \ldots attribute_n$, $attribute_k \ldots attribute_p$. If there is a great number of single alternatives, we propose a two-phase method in order to carry out the Assessment phase in a more efficient way: (1) a selection of groups of alternatives that are considered for each criterion (Clustering of Alternatives), where the elements of these groups present similar behaviour according to the criterion; and (2) an automatic analysis of these groups of alternatives, which will permit the number of alternatives in the AHP to be reduced significantly (Disaggregation of Alternatives). These groups are subsets of BDCs and attributes as alternatives for each criterion.

For the example detailed in Section 3, the single alternatives are 16 (5 BDCs and 11 attributes). In Table 4, the 'simplified' alternatives are shown (clusters of alternatives for criteria) established by the business expert in order to simplify the hierarchy.

Once the criteria have been defined, and their simplified alternatives are selected, the next step entails the pairwise comparison of simplified alternatives for their evaluation regarding each criterion. Simplified alternatives are used so that the business expert can easily establish the numerical values in the pairwise comparisons. For our example, the pairwise comparisons of alternatives per criteria are shown in Fig. 8. For example, for criterion c1, a fault probability is higher for alternative $SA1_{c1}$ than for alternative $SA2_{c1}$, and therefore the cell $c1[SA2_{c1}, SA1_{c1}]$ is 4, and $c1[SA1_{c1}, SA2_{c1}]$ is 1/4. In other words, a fault probability in $SA2_{c1}$ is 4 times less probable than a fault in $SA1_{c1}$.

### 5.4. *Assessment phase*

In this phase, a set of pairwise comparison matrices are filled out. Also, this phase allows consistency checks of pairwise values through a

**Table 4**
Alternatives for criteria.

| | |
|---|---|
| | Criterion c1: Fault probability |
| SA1$_{c1}$ | softwareCost, hardwareCost, humanCost, subsidisedCost, BDC$_1$ |
| SA2$_{c1}$ | BDC$_2$, BDC$_5$ |
| SA3$_{c1}$ | totalCost, subsidisedPerYear, humanCostPerYear, maximumSubsidised, potentialSubsidised, maximumHardware, maximumHuman, BDC$_3$, BDC$_4$ |
| | Criterion c2: Human reliability |
| SA1$_{c2}$ | softwareCost, hardwareCost, humanCost, subsidisedCost |
| SA2$_{c2}$ | subsidisedPerYear, humanCostPerYear |
| SA3$_{c2}$ | totalCost, potentialSubsidised, maximumHardware, maximumSubsidised, maximumHuman |
| | Criterion c3: Source reliability |
| SA1$_{c3}$ | softwareCost, hardwareCost, humanCost, subsidisedCost, subsidisedPerYear, humanCostPerYear |
| SA2$_{c3}$ | BDC$_1$, BDC$_2$, BDC$_3$, BDC$_4$, BDC$_5$ |
| SA3$_{c3}$ | totalCost, potentialSubsidised, maximumHardware, maximumHuman, maximumSubsidised |
| | Criterion c4: Data accuracy |
| SA1$_{c4}$ | softwareCost, hardwareCost, humanCost, subsidisedCost |
| SA2$_{c4}$ | maximumSubsidised, maximumHuman, totalCost, potentialSubsidised, maximumHardware |
| SA3$_{c4}$ | subsidisedPerYear, humanCostPerYear |
| | Criterion c5: BDC persistence |
| SA1$_{c5}$ | BDC$_1$ |
| SA2$_{c5}$ | BDC$_2$, BDC$_5$ |
| SA3$_{c5}$ | BDC$_3$, BDC$_4$ |

consistency index. The business expert can easily establish the numerical values in the comparisons using simplified alternatives, since each criterion may naturally lead the expert to define certain alternatives that are different for each criterion. However, in order to establish weights for the final AHP, global alternatives are needed that are valid for all defined criteria.

The simplified alternatives are split into as many global alternatives as necessary for all the elements in the same global alternative to present the same behaviour regarding each defined criterion. This may give rise to a large number of global alternatives, each of which is composed of few elements. However, they are transparent to the user and/or the expert, since simplified alternatives are automatically transformed into global alternatives and processed in subsequent steps of the methodology. As a result, the global alternatives and their corresponding evaluations per criterion are obtained automatically.

In this way, for our example, the defined simplified alternatives give rise to seven global alternatives:

A1: softwareCost, hardwareCost, subsidisedCost, humanCost
A2: subsidisedPerYear, humanCostPerYear
A3: potentialSubsidised, maximumHardware, totalCost maximumSubsidised, maximumHuman
A4: BDC$_3$, BDC$_4$
A5: BDC$_2$
A6: BDC$_1$
A7: BDC$_5$

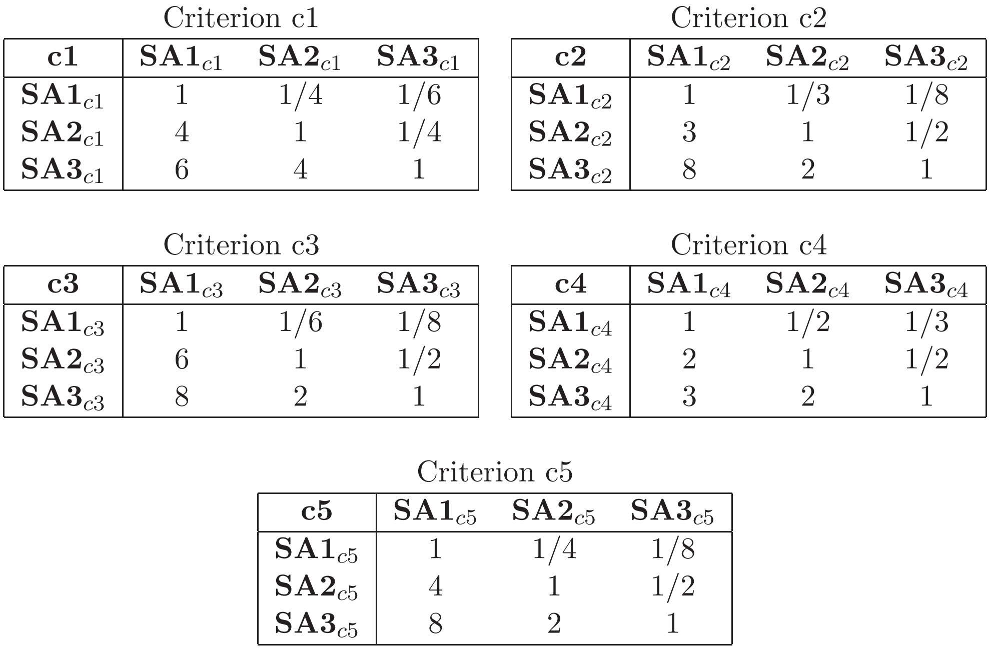
These 7 global alternatives instead of 16 single alternatives (5 BDCs and 11 attributes) thereby imply a reduction of almost 55% of the alternatives to be considered.

Likewise, in order to illustrate the automatic generation and filling in of comparison matrices, the matrix for criterion $c1$ is shown in Table 5. The cells compare two global alternatives Ax and Ay for the criterion c: the alternatives come from the same simplified alternative for c, and their evaluation is always 1 (the cell is filled in with value 1).

For example, for criterion c1, the global alternatives A2 and A3 come from the same simplified alternative SA3$_{c1}$, and therefore the cells c1[A2, A3] and c1[A3, A2] in Table 5 obtain the value 1. Furthermore, this also occurs when comparing alternatives which are not evaluable through a particular criterion. For example, if the criterion c is only defined for data, then the alternatives that include BDCs are always evaluated to value 1 when compared with alternatives that include data.

### 5.5. Prioritisation Phase: Priorities of the alternatives

In this phase, the priorities obtained from the pairwise comparisons are transformed into a priority for each alternative. Once the global alternatives have been obtained, then automatic mathematical



Criterion c1

| c1 | SA1$_{c1}$ | SA2$_{c1}$ | SA3$_{c1}$ |
|---|---|---|---|
| SA1$_{c1}$ | 1 | 1/4 | 1/6 |
| SA2$_{c1}$ | 4 | 1 | 1/4 |
| SA3$_{c1}$ | 6 | 4 | 1 |

Criterion c2

| c2 | SA1$_{c2}$ | SA2$_{c2}$ | SA3$_{c2}$ |
|---|---|---|---|
| SA1$_{c2}$ | 1 | 1/3 | 1/8 |
| SA2$_{c2}$ | 3 | 1 | 1/2 |
| SA3$_{c2}$ | 8 | 2 | 1 |

Criterion c3

| c3 | SA1$_{c3}$ | SA2$_{c3}$ | SA3$_{c3}$ |
|---|---|---|---|
| SA1$_{c3}$ | 1 | 1/6 | 1/8 |
| SA2$_{c3}$ | 6 | 1 | 1/2 |
| SA3$_{c3}$ | 8 | 2 | 1 |

Criterion c4

| c4 | SA1$_{c4}$ | SA2$_{c4}$ | SA3$_{c4}$ |
|---|---|---|---|
| SA1$_{c4}$ | 1 | 1/2 | 1/3 |
| SA2$_{c4}$ | 2 | 1 | 1/2 |
| SA3$_{c4}$ | 3 | 2 | 1 |

Criterion c5

| c5 | SA1$_{c5}$ | SA2$_{c5}$ | SA3$_{c5}$ |
|---|---|---|---|
| SA1$_{c5}$ | 1 | 1/4 | 1/8 |
| SA2$_{c5}$ | 4 | 1 | 1/2 |
| SA3$_{c5}$ | 8 | 2 | 1 |

**Fig. 8.** Comparisons of alternatives from criterion c1 to c5.

**Table 5**
Comparisons of global alternatives for criterion $c1$.

| c1 | A1 | A2 | A3 | A4 | A5 | A6 | A7 |
|---|---|---|---|---|---|---|---|
| A1 | 1 | 1/6 | 1/6 | 1/6 | 1/4 | 1 | 1/4 |
| A2 | 6 | 1 | 1 | 1 | 4 | 6 | 4 |
| A3 | 6 | 1 | 1 | 1 | 4 | 6 | 4 |
| A4 | 6 | 1 | 1 | 1 | 4 | 6 | 4 |
| A5 | 4 | 1/4 | 1/4 | 1/4 | 1 | 4 | 1 |
| A6 | 1 | 1/6 | 1/6 | 1/6 | 1/4 | 1 | 1/4 |
| A7 | 4 | 1/4 | 1/4 | 1/4 | 1 | 4 | 1 |

processing of the comparisons is applied using the existing methods (Bouyssou, Marchant, Pirlot, Tsoukiás, & Vincke, 2006) in order to derive priorities for each global alternative. For our example, after mathematical processing, the priorities obtained for the global alternatives are shown in Table 6.

### 5.6. Creation of the COP in accordance with the priorities

A priority is obtained for each alternative in the Prioritisation Phase. These priorities are transformed into weights, and for each priority an integer number is obtained. Each priority is multiplied by 100 and is rounded up to the nearest integer.

The objective function is transformed in order to include these weights:

$$minimize\left(\sum_{j=1}^{m}(rV_j \cdot wd(V_j)) + \sum_{i=1}^{n}(rBDC_i \cdot minLik_i \cdot wb(BDC_i))\right)$$

The function $wd(V_j)$ returns the weight associated to the variable $V_j$, and the function $wb(BDC_i)$ returns the weight associated to $BDC_i$. Fig. 9 shows the application of these priorities to the objective function for the proposed example. The differences, for the proposed example, between the Min-CSP with priorities and that without priorities (Fig. 6) are shown in italics in Fig. 9.

The minimal diagnosis for the example presented in Fig. 6 is a set of three elements that must be modified:

- The value associated with the variable *maximumHuman*1 (used in some instances of $BDC_4$) is erroneous, the correct value is 23,000.
- $BDC_5$ is not correct: the correct BDC should be the expression $\mathit{subsidisedCost} \geqslant 2 \cdot \mathit{subsidisedPerYear}$.
- Finally, the value associated with *subsidisedPerYear*5 (used in one instance of $BDC_3$) is erroneous. Note that the variable *potentialSubsidised*5 appears in the minimal diagnosis obtained in Section 4.3, but it does not appear here because priorities are used. The variable *potentialSubsidised*5 (attribute *potentialSubsidised*) is included in alternative A3 (which has a priority 0.2), and *subsidisedPerYear*5 is included in alternative A2 (which has a priority 0.18). Therefore, a review of *subsidisedPerYear*5 is preferred to that of *potentialSubsidised*5.

## 6. Evaluation

Previously, a reduced part of the real example has been used to facilitate the understanding. In this section, a more complex example is tackled. The relational model is depicted in Fig. 10 and the BDCs are:

1. $\mathit{subsidisedCost} \geqslant 2*\mathit{subsidisedPerYear}$
2. $\mathit{humanCost} \geqslant 2*\text{humanCostPerYear}$
3. $\mathit{subsidisedPerYear} \leqslant \mathit{maximumSubsidised}$
4. $\mathit{humanCostPerYear} \leqslant \mathit{maximumHuman}$
5. $\mathit{humanCost} * 3 \geqslant \mathit{totalCost}$
6. $\mathit{hardwareCost} + \mathit{softwareCost} + \mathit{humanCost} = \mathit{totalCost}$
7. $\mathit{softwareCost} < \mathit{humanCost}$
8. $\mathit{humanCost} \geqslant 2 * \mathit{hardwareCost}$
9. $\mathit{subsidisedPerYear} \leqslant \mathit{potentialSubsidised}$
10. $\mathit{humanCost} \geqslant \mathit{potentialSubsidised}$
11. $\mathit{maximunIncentivable} \geqslant \mathit{potentialSubsidised}$
12. $\mathit{subsidisedPerCompany} \leqslant \mathit{subsidisedPerYear}$
13. $3 * \mathit{hardwareCost} > \mathit{subsidisedPerCompany}$
14. $\mathit{reducedQuantity} \leqslant \mathit{maxReducedQuantity}$
15. $\mathit{reducedQuantity} \geqslant \mathit{minReducedQuantity}$

**Table 6**
Priorities of the global alternatives.

| A1 | A2 | A3 | A4 | A5 | A6 | A7 |
|---|---|---|---|---|---|---|
| 0.10 | 0.18 | 0.20 | 0.20 | 0.11 | 0.12 | 0.09 |

The example contains 15 BDCs, and there are 864 instances of these BDCs where the variables store 682 different values. To carry out the evaluation of this proposal, a set of experiments has been designed to simulate possible multiple and simple faults, both in data and in BDCs. The use of these tests has allowed us to confirm the applicability and validity of our method, since it obtains the minimum diagnosis in most cases. The created COPs have been solved using ILOG solver $^{TM}$.

In a first step, $BDC_{10}$ is changed and nine incorrect (and random) values are stored in nine variables: *subsidisedPerYear20*, *subsidisedPerYear31*, *subsidisedPerCompany10*, *subsidisedPerCompany58*, *humanCost9*, *humanCost12*, *totalCost9*, *totalCost11*, *totalCost19*. The results after the valuation of the variables in the BDCs are shown in Table 7. The analysis of each BDC according to the tuples yields a result of 61 incorrect instances.

A Min-CSP is created for the diagnosis of the example. A total of 336 minimal diagnoses are obtained solving this COP, which entails combinations of 10 elements. In order to satisfy all the instances, a total of 10 modifications must therefore be made. These 10 modifications imply: a change in $BDC_{10}$, a change of the value associated with variables *humanCost*12, *subsidisedPerYear*20 and *subsidisedPerYear*31, and the following 6 modifications:

- A change in one variable of the set: {*subsidisedPerCompany*10, *subsidisedPerYear*5}.
- A change in one variable of the set: {*subsidisedPerCompany*58, *subsidisedPerYear*29}.
- A change in one variable of the set: {*hardwareCost*11, *softwareCost*11, *totalCost*11}.
- A change in one variable of the set: {*hardwareCost*19, *softwareCost*19, *humanCost*19, *totalCost*19}.
- Two changes derived from one of the following two options:
  - A change of the value associated with variable *humanCost*9, and another change in one variable of the set: {*hardwareCost*9, *softwareCost*9, *totalCost*9}.
  - A change of the value associated with variable *humanCostPerYear*26, and another change in one variable of the set: {*hardwareCost*9, *softwareCost*9, *humanCost*9, *totalCost*9}.

There are several minimal diagnoses since it is possible to satisfy the 61 unsatisfied instances by changing one of the proposed minimal combinations.

In order to reduce the set of minimal diagnoses, an AHP is incorporated. The obtained priorities are used as weight values of the objective function. For our example, the ranking for global alternatives is shown in Table 8. By including the set of weights derived from the AHP and solving the Min-CSP problem, a total of 36 minimal diagnoses are obtained, thereby reducing the complexity of the solution. This set of minimal diagnoses is a subset of the initial set of 336 (when weights derived from the AHP are not used). These 36 minimal diagnoses are combinations of 10 modifications: a change in $BDC_{10}$, a change of the value associated with variables *subsidisedPerYear*20, *subsidisedPerYear*31, *subsidisedPerCompany*10, *subsidisedPerCompany*58,

**Variables of the problem:**

integer maximumHuman1, maximumHuman2, maximumHuman3, humanCostPerYear1, ...

integer[0,1] $rMh_1$, $rMh_2$, $rMh_3$, $rHpy_1$, $rHpy_2$, $rHpy_3$, $rHpy_4$, $rHpy_5$, ...

..., $rBDC_4$, $rBDC_4^1$, $rBDC_4^2$, $rBDC_4^3$, $rBDC_4^4$, $rBDC_4^5$, $rBDC_5$, $rBDC_5^1$, ..., $rBDC_5^5$

*integer[1,100]* $wMh_1 = 20$, $wMh_2 = 20$, $wMh_3 = 20$, $wHpy_1 = 18$, $wHpy_2 = 18, \ldots,$

$wBDC_1 = 12$, $wBDC_2 = 11$, $wBDC_3 = 20$, $wBDC_4 = 20$, $wBDC_5 = 9$

**Constraints related to the instances of variables:**

$rMh1 = \neg(maximumHuman1 = 13000)$ $rMh2 = \neg(maximumHuman2 = 24000)$

...

$rHpy1 = \neg(humanCostPerYear1 = 23000)$ $rHpy2 = \neg(humanCostPerYear2 = 22000)$

...

**Constraints related to the instances of BDCs:**

...

$rBDC_4^1 = \neg(humanCostPerYear1 \leq maximumHuman1)$

$rBDC_4^2 = \neg(humanCostPerYear2 \leq maximumHuman2)$

$rBDC_4^3 = \neg(humanCostPerYear3 \leq maximumHuman1)$

$rBDC_4^4 = \neg(humanCostPerYear4 \leq maximumHuman2)$

$rBDC_4^5 = \neg(humanCostPerYear5 \leq maximumHuman3)$

...

$rBDC_4 = (rBDC_4^1 + rBDC_4^2 + rBDC_4^3 + rBDC_4^4 + rBDC_4^5 \geq minLik_4)$

...

$(rBDC_4^1 + rBDC_4^2 + rBDC_4^3 + rBDC_4^4 + rBDC_4^5 = 0) \vee$
$(rBDC_4^1 + rBDC_4^2 + rBDC_4^3 + rBDC_4^4 + rBDC_4^5 \geq minLik_4)$

...

**Objective function:**

$minimize(rMh_1 \cdot wMh_1 + rMh_2 \cdot wMh_2 + rMh_3 \cdot wMh_3 + rHpy_1 \cdot wHpy_1 + rHpy_2 \cdot wHpy_2 + \ldots$
$\ldots + rBDC_4 \cdot minLik_4 \cdot wBDC_4 + rBDC_5 \cdot minLik_5 \cdot wBDC_5)$

**Fig. 9.** Min-CSP of the example including priorities.

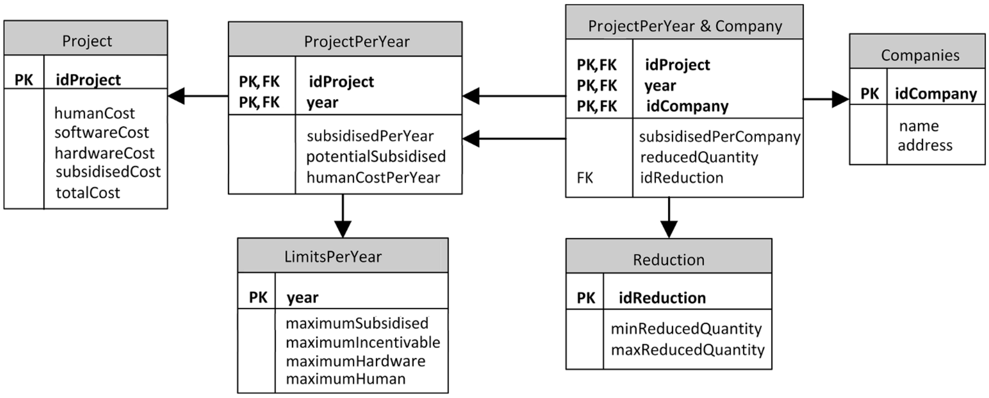


**Fig. 10.** Relational Model of the extended example.

**Table 7**
Results after the valuation of the BDCs.

| Non-satisfied instances | BDCs |
|---|---|
| 0 ✓ | $BDC_1$, $BDC_3$, $BDC_4$, $BDC_9$<br>$BDC_{11}$, $BDC_{13}$, $BDC_{14}$, $BDC_{15}$ |
| 1 × | $BDC_2$, $BDC_5$, $BDC_7$, $BDC_8$ |
| 4 × | $BDC_6$ |
| 6 × | $BDC_{12}$ |
| 47 × | $BDC_{10}$ |

**Table 8**
Priorities of the global alternatives.

| | **AHP** |
|---|---|
| $BDC_1$, $BDC_2$, $BDC_7$, $BDC_8$ | 0.06 |
| $BDC_5$, $BDC_{10}$, $BDC_{11}$, $BDC_{12}$, $BDC_{13}$ | 0.07 |
| humanCost, hardwareCost, softwareCost, totalCost, subsidisedCost, reducedQuantity | 0.07 |
| $BDC_6$, $BDC_9$ | 0.08 |
| subsidisedPerCompany, maximumIncentivable | 0.10 |
| subsidisedPerYear, humanCostPerYear, potentialSubsidised | 0.11 |
| maxReducedQuantity, minReducedQuantity, maximumSubsidised, maximumHuman | 0.12 |
| $BDC_3$, $BDC_4$, $BDC_{14}$, $BDC_{15}$ | 0.14 |

*humanCost*9 and *humanCost*12, and the following 3 modifications:

- A change in one variable of the set: {*hardwareCost*9, *softwareCost*9, *totalCost*9}.
- A change in one variable of the set: {*hardwareCost*11, *softwareCost*11, *totalCost*11}.
- A change in one variable of the set: {*hardwareCost19*, *softwareCost19*, *humanCost19*, *totalCost11*}.

The minimal diagnosis depends on the number of faults, and on the number of variables and tuples affected by these faults. To obtain a complete set of tests which covers the different types of faults, the variables and BDCs have been divided into various sets. Regarding the instances of the variables (682 in the example), 25 sets of variables were created. The criterion for these sets was the number of tuples and the set of affected BDCs. For example, the variable *subsidisedCost*8 appears in four tuples affected by the same BDC, and the variable *hardwareCost*20 appears in four tuples of three different BDCs. The BDCs were grouped into 8 sets, whose criterion was the number of affected variables. For example, 40 variables are affected by $BDC_5$, $BDC_7$, and $BDC_8$, and 61 variables are affected by $BDC_3$, $BDC_4$, and $BDC_{11}$.

Fig. 11 details the execution time (in milliseconds) on a logarithmic scale with base 10. Each column is named with two numbers: the first number represents the counter of erroneous BDCs, and the second number represents the counter of erroneous input data. For each column, five tests are executed by selecting different types of variables and BDCs. These tests are shown as a box and whisker chart. A total of 96 tests are represented in Fig. 11. For all these single tests, the greatest time consumed is 21, 234 milliseconds, and the minimal is 578 milliseconds. The time consumed for the minimal diagnosis calculation depends on the number of non-satisfied BDCs instances.

## 7. Limitations

The limitations in these papers are related to the knowledge of the organisation regarding the relationship between their data, the capacity to model the data behaviour, the complexity in computing the COPs, and the vast quantity of data and business rules that can explain a malfunction. Aspects of these topics include:

- The **knowledge concerning the possible values of the attributes** managed during the business process instance. If the possible values of the variables and their relations are unknown, it will not be possible to determine a malfunction or to prioritise a set thereof. This means that for the priority-based diagnosis proposed in this paper, a key aspect is that part of the behaviour of the company is known using Business Rules. It is also related to the semantic capacity for representing the data relation, according to the real scenarios to be modelled. If business rules cannot be modelled as regular expressions of a grammar, then the model can not represent the input data relation.
- The **COP time evaluation** is frequently a handicap for Constraint Programming solvers. Since the problem is solved as a COP, the validation time depends on the complexity of constraints and the domain of the variables. The complexity of CSP problems has been analysed in recent decades (Cheeseman, Kanefsky, & Taylor, 1991). Several considerations and the analysis of how the complexity can be affected by Constraint Satisfaction solvers in business processes are included in Gómez-López et al. (2014). In order to demonstrate the usability of our proposal, Section 6 addresses a real scenario with a significant number of variables and BDCs to show the range of the evaluation times.
- When a **vast quantity of data and BDCs** is computed in a diagnosis process, several explanations can be found. The AHP can drastically reduce this number, but it can be remain high. As analysed in Section 6, the computation time can increase, and the time needed to study the possibilities can also be high, although approximation does indeed improve and facilitate the analysis of the possible errors that bring out faults in a system. As long as there are more non-satisfied instances, then the more complex it becomes to find a minimal diagnosis that satisfies all instances. Therefore, it is important to apply the methodology when BDCs are designed, or when a BDC is changed, in order to prevent the accumulation of errors.

## 8. Conclusions and Future work

We propose a full methodology for the identification of the causes of non-conformance between Data and Business Rules (Business Data Constraints). In this paper, a full methodology carries out a comparison of the priority of the components, and executes the automatic diagnosis process, which includes the likelihood of errors in each item of data and each BDC to ascertain the most likely elements that can be failing. Our approach includes the likelihood of errors in data and BDCs in order to determine the most promising candidate that is responsible for a fault.

The Analytic Hierarchy Process (AHP) is applied to obtain the probabilistic constraints according to the priority description regarding faults described by the users and/or experts. Moreover, since this process may become tedious when dealing with a large quantity of data and/or BDCs in the model, we provide an innovative AHP clustering process in order to facilitate this task. The AHP clustering process

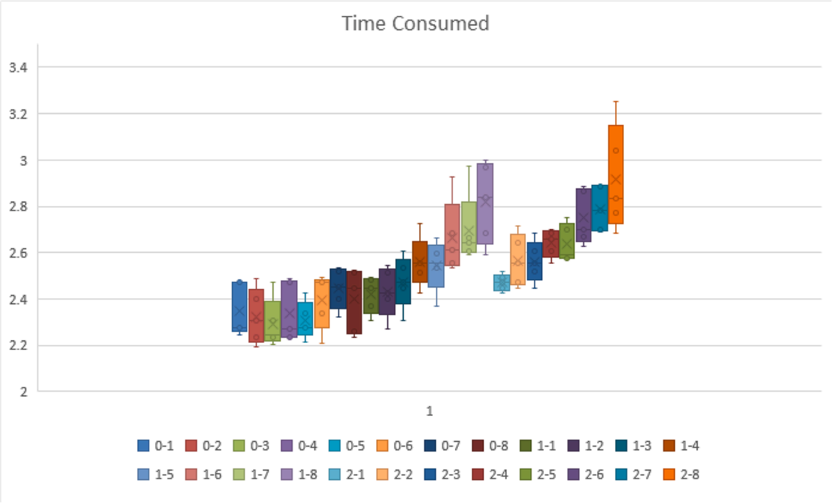


**Fig. 11.** The execution time of test cases by applying AHP.

simplifies the prioritisation of alternatives for the users and/or experts (by defining simplified alternatives and then global alternatives), whereas the rest of the process is automated (the problem is modelled and solved as a COP). To evaluate our approach, a complex real example has been given in order to simulate single and multiple faults in data and BDCs.

Furthermore, as future work, our main line of research involves storing the previous diagnosis and the real errors in order to update the weights used for prioritisation and to prevent future faults by using techniques to prognosticate. Another interesting line of study involves an extension of the proposed grammar to enable the incorporation of aggregate data.

## CRediT authorship contribution statement

**Diana Borrego:** Conceptualization, Data curation, Methodology, Resources, Software, Writing - original draft, Writing - review & editing. **María Teresa Gómez-López:** Conceptualization, Data curation, Methodology, Supervision, Validation, Writing - original draft, Writing - review & editing. **Rafael M. Gasca:** Conceptualization, Data curation, Methodology, Supervision, Validation, Writing - original draft, Writing - review & editing.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

This work has been partially funded by the Ministry of Science and Technology of Spain by the ECLIPSE projects (RTI2018–094283-B-C33) and the European Regional Development Fund (ERDF/FEDER).

## References

Awad, A., Weidlich, M., & Weske, M. (2011). Visually specifying compliance rules and explaining their violations for business processes. *Journal of Visual Languages and Computing, 22*(1), 30–55. https://doi.org/10.1016/j.jvlc.2010.11.002

Becker, J., Ahrendt, C., Coners, A., Weiß, B., & Winkelmann, A. (2011). Modeling and analysis of business process compliance. In M. Nüttgens, A. Gadatsch, K. Kautz, I. Schirmer, & N. Blinn (Eds.), *Governance and Sustainability in Information Systems, volume 366 of IFIP Publications* (pp. 259–269). Springer. https://doi.org/10.1007/978-3-642-24148-2_17.

Beheshti, S.-M.-R., Benatallah, B., Sakr, S., Grigori, D., Motahari-Nezhad, H. R., Barukh, M. C., Gater, A., & Ryu, S. H. (2016). Process Analytics - Concepts and Techniques for Querying and Analyzing Process Data. *Springer*. https://doi.org/10.1007/978-3-319-25037-3

Borrego, D., Eshuis, R., Gómez-López, M. T., & Gasca, R. (2013). Diagnosing correctness of semantic workflow models. *Data Knowledge Engineering, 87*, 167–184. https://doi.org/10.1016/j.datak.2013.04.008

Borrego, D., Gasca, R., & Gómez-López, M. (2015). Automating correctness verification of artifact-centric business process models. *Information and Software Technology, 62*, 187–197. https://doi.org/10.1016/j.infsof.2015.02.010

Borrego, D. & Gómez-López, M.T. (2019). Diagnosing business processes. In B.P.V.P. eds. T. Escobet, A. Bregon (Ed.), Fault Diagnosis of Dynamic Systems: Quantitative and Qualitative Approaches, 389–408, ISBN: 9783030177270 (pp). Cham: Springer Nature Switzerland.

Bouyssou, D., Marchant, T., Pirlot, M., Tsoukiás, A., & Vincke, P. (2006). Evaluation and decision models with multiple criteria: Stepping stones for the analyst. International Series in Operations Research and Management Science, Volume 86. Boston, 1st edition. doi:10.1007/0-387-31099-1.

Ceballos, R., Borrego, D., López, M.T.G., & Gasca, R.M. (2016). 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, June 13–14, 2016, Proceedings (pp. 212–227). doi:10.1007/978-3-319-39429-9_14.

Cheeseman, P., Kanefsky, B., & Taylor, W. M. (1991). In *Where the really hard problems are In Proceedings of the 12th International Joint Conference on Artificial Intelligence* (Volume 1, pp. 331–337). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.. IJCAI'91.

El-Qurna, J., Yahyaoui, H., & Almulla, M. (2017). A new framework for the verification of service trust behaviors. *Knowledge-Based Systems, 121*, 7–22. https://doi.org/10.1016/j.knosys.2017.01.011

Eshuis, R., & Kumar, A. (2010). An integer programming based approach for verification and diagnosis of workflows. *Knowledge-Based Systems, 69*(8), 816–835. https://doi.org/10.1016/j.datak.2010.03.003

Gómez-López, M. T., Gasca, R. M., & Pérez-Álvarez, J. M. (2014). Decision-making support for the correctness of input data at runtime in business processes. *International Journal of Cooperative Information Systems, 23*(4). https://doi.org/10.1142/S0218843014500038

Gómez-López, M. T., Gasca, R. M., & Pérez-Álvarez, J. M. (2015). Compliance validation and diagnosis of business data constraints in business processes at runtime. *Information Systems, 48*, 26–43. https://doi.org/10.1016/j.is.2014.07.007

Knuplesch, D., & Reichert, M. (2017). A visual language for modeling multiple perspectives of business process compliance rules. *Software and Systems Modeling, 16*, 715–736. https://doi.org/10.1007/s10270-016-0526-0

Knuplesch, D., Reichert, M., & A., K. (2017). A framework for visually monitoring business process compliance. Information Systems, 64, 381–409. doi:10.1016/j.is.2016.10.006.

Köksalan, M., Mousseau, V., Ozpeynirci, O., & Bilgin Ozpeynirci, S. (2009). An outranking-based approach for assigning alternatives to ordered classes. *Naval Research Logistics, 56*(1), 74–85.

Ly, L., Rinderle-Ma, S., Knuplesch, D., & Dadam, P. (2011). Monitoring business process compliance using compliance rule graphs. In In Meersman, R., et al. (eds.) OTM 2011, Part I., LNCS, vol. 7044 (pp. 82–99). doi:10.1007/978-3-642-25109-2_7.

Maggi, F., Montali, M., & van der Aalst, W. (2012). An operational decision support framework for monitoring business constraints. *LNCS, 7212*, 146–162. https://doi.org/10.1007/978-3-642-28872-2_11

Montali, M., Maggi, F., Chesani, F., Mello, P., & van der Aalst, W. (2013). Monitoring business constraints with the event calculus. *ACM TIST, 5*(1). https://doi.org/10.1145/2542182.2542199

Peng, Y., & Reggia, J. (1990). *Abductive Inference Models for Diagnostic Problem-solving. Symbolic computation*. Springer.

Pereira, F., Figueira, J., Mousseau, V., & Roy, B. (2009). Comparing two territory partitions in districting problems: Indices and practical issues. *Socio-Economic Planning Sciences, 43*(1), 72–88. https://doi.org/10.1016/j.seps.2007.04.001

Pérez-Álvarez, J. M., López, M. T. G., Eshuis, R., Montali, M., & Gasca, R. M. (2020). Verifying the manipulation of data objects according to business process and data models. *Knowledge and Information Systems, 62*(7), 2653–2683.

Ramos, B., mez López, M.T.G., Borrego, D., Ceballos, R., Gasca, R.M., & Barea, A. (2021). Self-adaptative troubleshooting for to guide resolution of malfunctions in aircraft manufacturing. IEEE Access, 10, 42707–42723.

Rossi, F., van Beek, P., & Walsh, T. (2006). *Handbook of Constraint Programming*. Elsevier.

Saaty, T. (2000). Fundamentals of the Analytic Hierarchy Process. RWS Publications, 4922 Ellsworth Avenue, Pittsburgh, PA 15413.

Saaty, T. L. (2008). Decision making with the analytic hierarchy process. *International Journal of Services Sciences*, 83–98. https://doi.org/10.1504/IJSSCI.2008.017590

Sidorova, N., Stahl, C., & Trcka, N. (2011). Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible. *Information Systems, 36*(7), 1026–1043. https://doi.org/10.1016/j.is.2011.04.004

Sun, S., Zhao, J., Nunamaker, J., & Sheng, O. (2006). Formulating the data-flow perspective for business process management. *Information Systems Research, 17*(4), 374–391. https://doi.org/10.1287/isre.1060.0105

Trcka, N., van der Aalst, W., & Sidorova, N. (2009). Data-flow anti-patterns: discovering data-flow errors in workflows. In van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009, LNCS vol. 5565 (pp. 425–439). doi:10.1007/978-3-642-02144-2_34.

van der Aalst, W.M.P., ter Hofstede, A.H.M., & Weske, M. (2003). Business process management: A survey. In Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26–27, 2003, Proceedings (pp. 1–12). doi:10.1007/3-540-44895-0_1.

Voglhofer, T., & Rinderle-Ma, S. (2020). Collection and elicitation of business process compliance patterns with focus on data aspects. *Business & Information Systems Engineering, 62*(4), 361–377.

Weidlich, M., Ziekow, H., Mendling, J., Günther, O., Weske, M., & Desai, N. (2011). Event-based monitoring of process execution violations. In: Rinderle-Ma, S. and Toumani, F. and Wolf, K. (eds.) BPM 2011, LNCS, vol. 6896 (pp. 182–198). doi:10.1007/978-3-642-23059-2_16.

Zyoud, S. H., & Fuchs-Hanusch, D. (2017). A bibliometric-based survey on AHP and TOPSIS techniques. *Expert Systems with Applications, 78*, 158–181. https://doi.org/10.1016/j.eswa.2017.02.016