Journal of Manufacturing Systems 47 (2018) 115–127

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

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

Knowledge-based multi-agent system for manufacturing problem solving process in production plants

Alvaro Camarillo[a,b,*], José Ríos[b], Klaus-Dieter Althoff[c,d]

[a] Exide Technologies GmbH, Odertal 35, 37431, Bad Lauterberg, Germany
[b] Mechanical Eng. Dept., Polytechnic University of Madrid, Jose Gutierrez Abascal 2, 2800, Madrid, Spain
[c] German Research Center for Artificial Intelligence (DFKI), Trippstadter Straße 122, 67663, Kaiserslautern, Germany
[d] University of Hildesheim, Universitätsplatz 1, 31141, Hildesheim, Germany

ARTICLE INFO

Keywords:
Manufacturing Problem Solving (MPS)
Case-Based Reasoning (CBR)
Process Failure Mode and Effect Analysis (PFMEA)
Ontology
Product Lifecycle Management (PLM)

ABSTRACT

This paper proposes a novel approach to develop a production-oriented software system aimed to assist shop floor actors during a Manufacturing Problem Solving (MPS) process. The proposed system integrates the problem-solving method 8D, Process Failure Mode and Effect Analysis (PFMEA), Case-Based Reasoning (CBR), and Product Lifecycle Management (PLM). The system is based on an ontology that enhances and extends existing proposals to allow representing any type of manufacturing problem linked to production lines and reusing PFMEA analysis results. The architecture of the system is based on SEASALT (Shared Experience using an Agent-based System Architecture LayouT), which is a multi-case base domain-independent reasoning architecture for extracting, analyzing, sharing, and providing experiences. A proof of concept prototype was developed, implemented, and tested in a company. The results, which were collected in two different manufacturing plants of the company, show the feasibility of the proposed approach and validate the conceptual proposal presented in this paper.

1. Introduction

Analytical methods are typically applied to prevent failures during the design phase of manufacturing processes and machinery. PFMEA (Process Failure Mode and Effect Analysis) is a preventive technique that allows identifying potential failure modes of a process and the effects of such failures. It also allows assessing the criticality of these effects on the production process. From a conceptual perspective, PFMEA is a preventive technique that helps avoid the occurrence of problems during the execution of manufacturing processes [1]. Nevertheless, despite the application of such preventive techniques, unforeseen failures can still occur during the operation of manufacturing systems.

A failure is an event in which some part of the manufacturing system does not perform according to its operational specifications. As a consequence, production is disrupted and production targets may not be reached. The gap between the resulting state and the intended state is a production problem. When a production problem appears, a procedure to analyze the problem in detail and generate a solution is needed. Several systematic methods such as PDCA, OPDCA, DMAIC, PROACT, Shainin, Kepner-Tregoe, and Eight Disciplines (8D) have been developed with that aim in mind. Such methods are framed under Continuous Improvement Process (CIP) and Manufacturing Problem Solving (MPS) [2–6]. Arguably, 8D is the problem-solving method that is more oriented to the resolution of production problems. Developed by Ford Motor Company in the early 1990's to support their plants and suppliers in the problem solving activity, 8D comprises eight main steps: (1) definition of a team, (2) description of the problem, (3) definition of containment actions, (4) root cause analysis, (5) definition of potential corrective actions and verification of effectiveness, (6) introduction of corrective actions, (7) definition of preventive actions and lessons learned, and (8) congratulate the team [8]. The knowledge and experience of team members [9,10] are key elements to implement any problem-solving method. These methods provide a structured process

Abbreviations: 8D, Eight Disciplines; AML, Aras Markup Language; CBR, Case-Based Reasoning; CIP, Continuous Improvement Process; ECR, Engineering Change Request; GUI, Graphical User Interface; ID, IDdentification number; KM, Knowledge Management; LLS, Lessons Learned System; MES, Manufacturing Execution System; MPS, Manufacturing Problem Solving; OEE, Overall Equipment Effectiveness; OWL, Ontology Web Language; PFMEA, Process Failure Mode and Effect Analysis; PLC, Programmable Logic Controller; PLM, Product Lifecycle Management; PPR, Product, Process and Resource; PSS, Problem Solving Sheet; SEASALT, Shared Experience using an Agent-based System Architecture LayouT; XML, eXtensible Markup Language

* Corresponding author at: Exide Technologies GmbH, Odertal 35, 37431, Bad Lauterberg, Germany.
E-mail addresses: alvaro.camarillo@eu.exide.com, a.camarillo@alumnos.upm.es (A. Camarillo), jose.rios@upm.es (J. Ríos), klaus-dieter.althoff@dfki.de (K.-D. Althoff).

https://doi.org/10.1016/j.jmsy.2018.04.002
Received 8 October 2017; Received in revised form 20 March 2018; Accepted 5 April 2018
Available online 04 May 2018

to facilitate the improvement and finding of solutions. Nevertheless, although training is generally provided to team members, these methods only bring good results when they are driven by actors with enough experience who get additional support knowledge (e.g., provided by a software tool [9,10]). The literature also shows that the industrial application of PFMEA is complex, time consuming, and inefficient [11]. In addition, it provides a low outcome, its results are not revised during regular continuous-improvement activities, and there are issues to keep an efficient feedback [11]. Part of the problem with PFMEA relates to the fact that it is based on a spreadsheet approach, which makes it difficult to reuse results and identify similarities [11].

This paper proposes a Knowledge Management (KM) approach that aims to:

- Facilitate the reuse of PFMEA analysis results.
- Facilitate the capture and reuse of data and knowledge of manufacturing processes, at shop floor level, in any manufacturing plant, during daily MPS activities linked to the Overall Equipment Effectiveness (OEE) improvement. Among the different topics considered by OEE, the focus is set on quality issues with product and processes (i.e. quality claims and scrap), abnormal production speed, and breakdowns.
- Provide shop floor actors with a problem-solving software tool based on the 8D method, which can be used even by users with very low knowledge of the manufacturing system with which they work.
- Support manufacturing knowledge sharing and integration across different manufacturing plants.

The proposed KM approach comprises the integration of the 8D MPS method [8] with Case-Based Reasoning (CBR) [12,13] on an agent-based distributed architecture with a Product Lifecycle Management (PLM) system [14,15] and PFMEA [1]. The 8D method provides a structured way to guide the resolution of problems step by step. CBR is used as an artificial intelligence tool to search for similar manufacturing problem cases collected previously in multiple locations. PLM is used as a source of extended context information about Product-Process-Resources (PPR) that will enrich the similarity calculation of the CBR application. PFMEA is used as source of the initial set of cases to feed the CBR application. One main contribution of this work is the integration of these four techniques: 8D method, CBR, PLM, and PFMEA.

The reminder of this paper is structured as follows. Section 2 contains a literature review of the main topics related to this work: manufacturing problem knowledge representation, PLM and CBR. Section 3 discusses the created models, which are the basis of the ontological approach adopted to define the data structures needed to manage manufacturing problem knowledge. Section 4 describes the developed prototype application and its validation. The paper ends with the conclusions and future works.

2. State of the art

2.1. Representation of manufacturing problems and PFMEA

Manufacturing problems need to be described in a consistent and systematic way in order to allow for a common understanding by the MPS personnel and an appropriate processing by a software system. One way to address this need is by means of an ontology [16]. Literature shows proposals of different ontologies focused on manufacturing related issues [17,18]. Manufacturing is a very wide domain and, depending on the specific area of interest, ontologies comprise a variety of manufacturing related concepts.

Chungoora et al. [18] propose a manufacturing core ontology with a manufacturing planning orientation, which comprises concepts related to product design and manufacturing processes and resources. The main concepts are: PartFamily, DesignFunction, Feature, ProcessPlan, ManufacturingMethod, ManufacturingProcess, ManufacturingResource, and ManufacturingFacility.

When considering the representation of concepts dealing with MPS and PFMEA, Kamsu-Foguem et al. [19] propose an ontology based on conceptual graphs to formalize knowledge in an experience feedback process. However, PFMEA concepts are not supported. Experience feedback is considered a relevant approach to support MPS with lessons learned formalized knowledge. The ontology comprises the following main concepts: feedback_object, experience_element, experience, action and attributes. The ontology also includes their corresponding specializations: activity, product, process, resource, competency, solution, context, analysis, event, positive_event, and negative_event. Scippacercola et al. [20] propose the use of SysML to create a system model where artifacts contain FMEA related information. The FMEA information is represented as a description of the logical states of the input flows, blocks, and their corresponding constraints. A transformation translates the annotated FMEA SysML model elements into facts and rules of a Prolog base. A Prolog engine queries the created model to derive FMEA results. Ebrahimipour et al. [21], Dittmann et al. [22], and Mikos et al. [23] present three examples of ontologies to support FMEA concepts, with the aim of facilitating the reuse of information stored in FMEA analyses.

Ebrahimipour et al. [21] propose an upper ontology where three concepts related to FMEA information (i.e., deviation, cause and consequence) are modeled as an event and activities. A deviation is modeled as an event, which is the beginning of a consequence. A consequence is an activity. A cause is an activity that causes a deviation.

Dittmann et al. [22] propose a ROOT_CONCEPT class that is specialized into seven subclasses: FMEA, Component, Function, Failure_mode, Control_method, Risk_priority_number, and Containment_action. They also propose a set of relationships among the classes. For instance, the “fulfills_a_function” relationship relates a Component to a Function, and “has_failure_mode” relates a Function to a Failure_mode. The classes Component and Function have associated taxonomies.

Mikos et al. [23] use the standard SAE J1739 and AIAG FMEA Reference Manual to define a PFMEA ontology. In addition to the PFMEA concepts modeled as classes (LocationOfFailure, PotentialCausesOfFailure, PotentialEffectsOfFailure, EndEffect, LocalEffect, PotentialFailureMode, and FMEADescription), the ontology seems to support the concepts of product, process, and function.

After reviewing the mentioned ontologies, the works from Chungoora et al. [18] and Dittmann et al. [22] were taken as reference. Section 3.2 explains the selected concepts that were adopted and the new concepts that are proposed, which are part of the contribution of this work.

When analyzing a manufacturing problem, it is important to know the context where it happens. The context information can be structured into three main areas: product, process and resource (PPR). PLM systems are considered the main source of PPR information. Therefore, the ontology must consider concepts managed by such systems.

2.2. Product lifecycle management as data repository for manufacturing problem solving

A PFMEA analysis is performed in a specific process and, thus, the information to be used is restricted to the components of that process. The identified potential failure modes relate to a specific manufacturing context, i.e. process, process step, machine, tooling, process parameters, and product manufacturing feature. Therefore, it is necessary to compare the context where a manufacturing problem occurs with the context of each existing PFMEA analysis in order to evaluate their similarity. In that way, PFMEA knowledge can be reused to assist in the solution of a specific manufacturing problem. The context of the problem can be described in the form of PPR data, which sets clear differences among problems. Lundgren et al. [11] consider PFMEA as part of the quality assurance activities and end up using a similar approach.

The use of digital models (comprising product design, process design and manufacturing information) enables an approach where process planning and quality assurance can be performed in an integrated way and where already-created information can be reused [11]. PLM is the main software system to support interoperable PPR digital models.

The use of a PLM system, as a central repository of data and as part of a Lessons Learned System (LLS), is also the approach adopted by Bertin et al. [24,25]. Their work focuses on the capture and reuse of knowledge along the Engineering Change Request (ECR) process. A change request is considered as an event and comprises a description and the proposal of feasible solutions. Applying a problem-solving method, an event is treated as a problem and its description comprises the following main information: complexity, criticality, resolution process and involved resources, root causes with justification, solutions with rationale, implementation actions, and planning. The exploitation of the capitalized knowledge is based on the search of LLS cards stored in the PLM system. The approach aims to improve the decision-making process of technical staff leading ECR processes.

Having reviewed ways of representing a manufacturing problem and the main repository for manufacturing problem context data, the next section reviews CBR as the artificial intelligence tool to support the search for solutions in an MPS process.

2.3. Case-based reasoning and manufacturing problem solving

Having access to prior problem cases may help when trying to solve an unforeseen manufacturing problem. CBR is particularly applicable to problems where earlier cases are available. CBR is also useful when the reality under analysis is too complex and difficult to be represented in a single model [12]. This is the case of manufacturing, with a huge variety of production processes that are impacted by many types of machines, environments, materials, methods, and personnel. Therefore, similarity determination between the current problem and the problems contained in the case base is an important factor. The previously mentioned manufacturing problem context data, which is stored in the PLM system, is proposed to enhance the similarity calculation.

Note that the adoption of CBR to assist during MPS activities is complementary to the use of predictive methods. For instance, the application of machine learning techniques, based on the analysis of big amounts of collected data, to make predictions in process monitoring and predictive or condition-based maintenance [26,27].

Manufacturing, as an application domain, represents also a big challenge for CBR due to its vast extension and potential amount of cases. A single case base containing information related to failures from hundreds of different types of processes, coming from many different manufacturing lines and plants, would create serious problems of retrieval speed and maintainability. Multi-case base reasoning systems were created years ago to address these types of issues [28]. Among the different approaches, and due to its flexibility about knowledge modularization, SEASALT (Shared Experience using an Agent-based System Architecture LayouT) [13] was selected for this work. SEASALT is a domain-independent architecture for extracting, analyzing, sharing, and providing experiences. SEASALT is based on the CoMES (Collaborative Multi-Expert-System) approach [29].

A relevant example of the CBR application in problem solving is the work of Reuss et al. [30] in an aircraft diagnose and maintenance context. They proposed a multi-agent system to assist in finding the root cause of a fault and providing maintenance suggestions. The system is based on the SEASALT architecture and uses post-flight reports, which contain faults occurred during flights, as main data source.

Mikos et al. [23] proposed a slightly different approach to PFMEA knowledge sharing and reuse to support the resolution of problems. Their approach, similar to SEASALT, is also based on a multi-agent architecture. They use RacerPro as the description logic reasoning engine for the inference service and knowledge retrieval. RacerPro provides the language nRQL to query a knowledge base compliant with an OWL (Ontology Web Language) ontology. The knowledge base is populated with PFMEA analysis cases. Given a potential failure mode the system seeks to retrieve all the linked potential end effects of failure and the potential causes of failure present in the PFMEA knowledge based.

Based on the introduced theoretical background, the next section presents the proposed models used to represent knowledge about manufacturing problems and to develop a software system to support MPS at shop floor level. The proposal integrates 8D, Case-Based Reasoning (CBR), Product Lifecycle Management (PLM), and PFMEA. 8D is used as an MPS method [8]. CBR [12] on an agent-based distributed architecture [13] is the artificial intelligence tool in charge of searching for similar manufacturing problem cases. The PLM system [15] contains the source of extended context information to enrich the similarity calculation within the CBR application. Finally, PFMEA [1] is involved as a preventive technique to create an initial set of potential manufacturing problem cases. This approach provides a wider and improved approach to manufacturing problem solving when compared to works identified in the literature. The work of Mikos et al. [23] is a PFMEA oriented solution where a potential failure mode is used to search for potential causes. The case base is limited to PFMEA, shop floor level problems are not fed into the case base, manufacturing context information is not considered to identify similar cases and the connection with a PLM system is not addressed. The work of Reuss et al. [30] uses the same CBR agent architecture and focuses on aircraft maintenance where faults are documented in post-flight reports. The use of PFMEA or any other manufacturing related preventive technique is not considered to feed the case base. The use of extended context related information to enrich the similarity calculation is not addressed either. Therefore, the connection to an additional software system, such as a PLM system, to enrich the search for solutions is out of its scope.

3. Developed models

3.1. Manufacturing problem solving process model

Following Toyota's production philosophy, the approach adopted in this work considers that an MPS process is embedded within a Continuous Improvement Process (CIP) cycle. Within a CIP cycle, a shop floor employee may compare the current condition of a given manufacturing process with its defined target condition. When the target is not reached, there is a production problem [4]. That situation is the starting point to execute an MPS process. Following an MPS method, a team should analyze the problem. The 8D method is an MPS method widely applied in industry [7,8]. When aiming to develop a software system to assist along an MPS process, it seems logical to consider a method that is already known by the employees in order to prevent an initial user rejection of the system. That is precisely the reason for selecting the 8D method in this work. The aim of the software system is to support the MPS process by guiding the employees/users through the eight steps of the method, providing additional information about the problem under analysis and knowledge related to possible solutions, and collecting the knowledge created during the process execution for reuse purposes. Fig. 1 shows the proposed MPS process model, which comprises eight main steps:

1. The user introduces a basic description of the manufacturing problem. Taking the Kepner-Tregoe method [31] as reference, the user should provide an answer to the following basic questions: 'What?' (a brief description of the problem), 'When?' (date and time), 'How often? (frequency), 'Where?' (this question is divided into three different attributes related to the line and station where the problem happens, and the product that is being produced), 'Who?' (operator), and 'Why?' (a brief description of why it is a problem). 'What?' and 'Why?' are implemented as free-text fields that may contain extended information about the problem under analysis, which could be useful for other users in the future to understand it

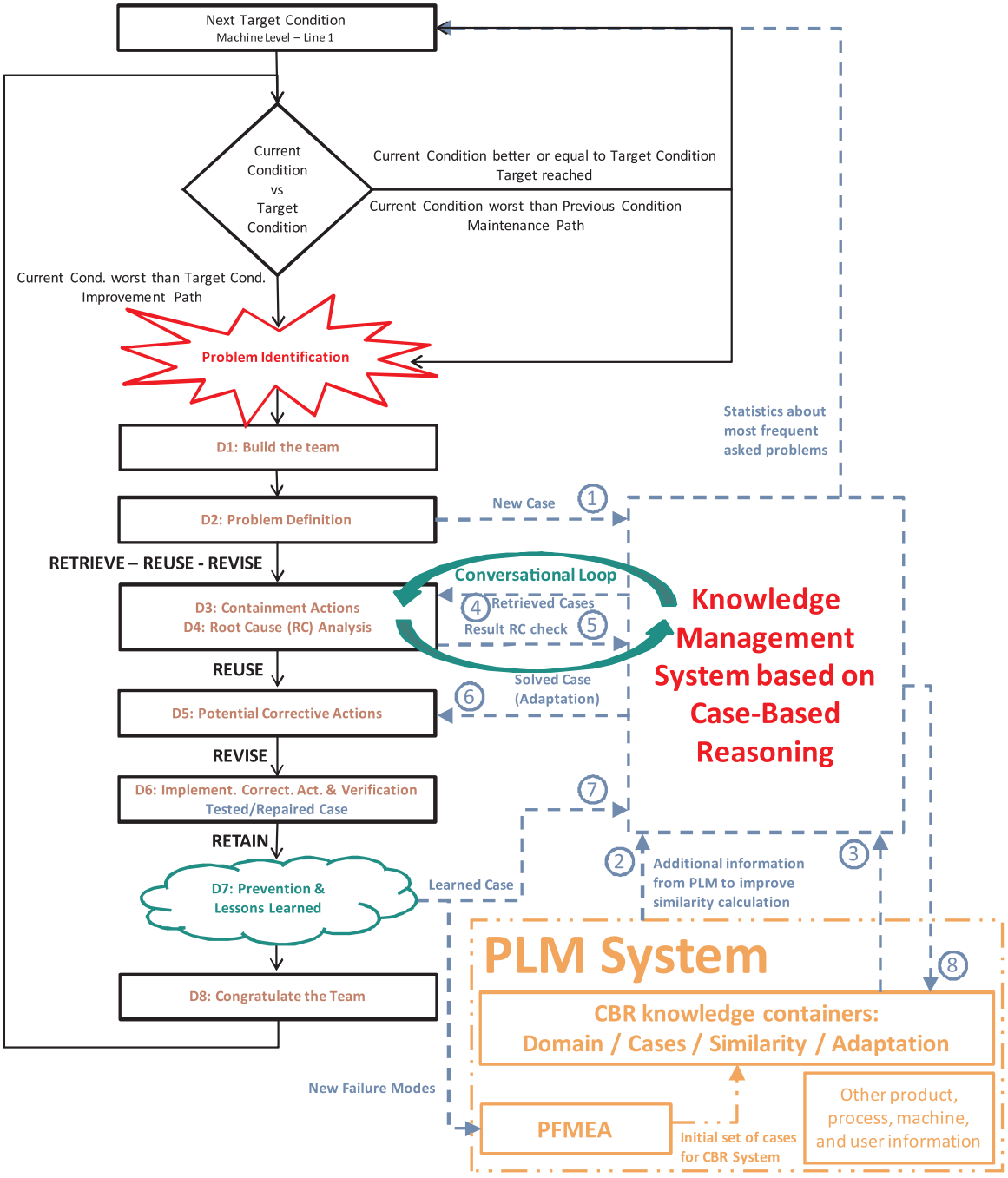

Fig. 1. Process model of the proposed system.

better. However, they are not used in the search for possible solutions. For the rest of questions and to ensure consistency and proper understanding between humans and machines, the Graphical User Interface (GUI) provides drop-down menus (Fig. 4). Where the user can select the most appropriate value. The available values are based on the ontology that will be presented below. The selected drop-down menus method also ensures that the time for the operator to give input to the system remains between one and two minutes maximum.

2. Given the answer to the basic questions, a subsystem searches and collects extended information related to the problem from the PLM system. The extended information comprises: process, machine, material, man, method, and environment. It is assumed that product manufacturing processes were digitally designed and that detailed digital information of processes and resources is stored in a PLM system (PPR data). This approach allows reusing manufacturing information that was created in the development phase to extend the information input by the user. This helps avoid situations where the user could not know all the details of the process due to a lack of experience. The output of the PLM comprises individual attribute values related to the problem context, for instance, process temperatures, process pressures, suppliers' name, and scrap level of the machine under evaluation (Fig. 5).
3. Combining the input from the user and the data collected from the PLM, a subsystem creates a global query to search for possible solutions. This global query is a list of attributes describing either the problem itself or the context of the problem. All these attributes are analyzed by the CBR subsystem. It then calculates their individual

similarity level relating the existing case base with manufacturing problem cases collected before. All calculated individual similarity values are merged in a global similarity result based on predefined system of weights, and this final value is used to identify the most similar cases that can be used as a solution proposal. The initial case base of the CBR system contains the result from PFMEA analysis conducted during the process development phase, and it is extended continuously later on in the step 8 of this MPS process. Sections 4.2, 4.3, and Fig. 6 explain with more detail all these sub-steps.

4. As a result, the CBR subsystem proposes possible containment actions and problem causes to the user (i.e., failure modes in the FMEA terminology).
5. The user must check the proposed failure modes or causes at the manufacturing line or station where the problem was identified and give feed-back to the system whether any of them is similar to the problem found. This step requires of a conversational CBR method. For this purpose, the approach from Becerra-Fernandez et al. [32] was selected. Following conversational CBR loops, failure cause after failure cause, the system guides the user down through the whole chain of causes until the very last root cause of the problem is identified. This step can take from a few minutes up to several hours. Its duration will depend on two factors. First, on how close to the real problem is the cases proposed as possible solution by the MPS system. Second, on the time needed by the operator to evaluate them in the machine.
6. Once the possible root causes are identified, the system provides the stored corrective and preventive actions. The user checks them and decides to apply, reject, or adapt them. This initial developed model does not consider an automatic adaptation process done by the CBR system itself, leaving this complex step of the CBR cycle [12] for future research. This step can take from just minutes (for easy repairs) up to months (when spare parts are not available or new devices have to be integrated in the machine to avoid quality issues).
7. As part of the lessons learned step, the user gives feedback to be analyzed by a Knowledge Engineer.
8. When appropriate, a Knowledge Engineer will update the CBR subsystem to extend the case base.

3.2. Manufacturing problem solving knowledge model

The proposed system is made of three main subsystems: an information input/output and coordination software unit, which provides the main Graphical User Interface (GUI), a PLM software application, and a CBR software application. The information to support the MPS process has to be managed by these three applications. The first step is to define which concepts are required by the MPS process.

This section presents a proposal for an ontology to represent MPS knowledge. Several constraints were considered in the development of the ontology. It should support any kind of manufacturing process and any location where a manufacturing process takes place. It should be compatible with the information structure of the PFMEA method. It should comprise concepts to describe different aspects of a manufacturing problem and concepts to be used for case similarity determination. A candidate concept for similarity determination requires to have a wide range of possible values depending on the manufacturing context to set differences among problems.

As it was mentioned in Section 2.1, the work from Dittmann et al. [22] was taken as the starting point. The proposed ontology takes the concepts: *Component*, *Function*, *Failure*, and some of their interrelationships. These three concepts will contain the core classification of a problem in the same format as a PFMEA [1], which will be used by the CBR system as main criteria to calculate similarities among problems. Dittmann et al. [22] proposed that a taxonomy should be associated to each of the three concepts, and consider any predefined one as a candidate. In the ontology proposed in this work, only the concepts of *Component* and *Function* have associated taxonomies, and both taxonomies correspond to the concepts defined by the PFMEA method [1]. The main differences with the approach from Dittmann et al. [22] are:

- The taxonomy of *Component* is flat and comprises six elements: *Process*, *Man*, *Machine*, *Material*, *Method*, and *Environment*.
- The taxonomy of *Function* has at its first level five elements representing the main functions that a component can fulfil: *Measure*, *Control*, *Make decision*, *Modify*, and *Be within specification*. Each of these five functions is extended into lower levels. The last element "Be within specification" is needed as type of function because the PFMEA method considers everything as components that fulfil functions, and then it helps to analyze the different ways in which a function can fail. In this sense, the requirement of being within a defined specification is considered from a functional point of view.
- The concept of *Failure* is modeled differently. A failure has the attribute '*Fulfilment rate*', which represents the grade of fulfilment of the associated function that the associated component is able to deliver in the problem under analysis. This attribute can get a value from a list of predefined values that starts at 0% and goes up to 200%.
- The concept *Problem* was added to the ontology to provide the link to a unique trio of *Component*, *Function* and *Failure*.
- The concept *Context* was added to consider the environment of the problem.

The following example aims to illustrate these basic concepts: a pneumatic cylinder (Machine) with a function to move something (Modify > Modify position > Transport) will have a Fulfilment rate of 0% if it doesn't move at all, or of 200% if it moves much faster than specified. Any value in between will represent a cylinder moving either slower or faster than specified. This is a subjective attribute, whose value depends on the perception of the employee when no quantitative measure is available. An employee may consider that a function is fulfilled at 25% while another may consider it fulfilled at 32% or at 23%. An exact number is not actually needed; instead, an approximate value to be used in the similarity calculation is all that is needed. For that reason, the allowed values of the attribute are defined as a list, which provides a guidance to the employee. The value 100% is not included in the list because this would mean that the function is fulfilled and thus there is no problem. The proposed ontology is represented using a lightweight UML class diagram (Fig. 2).

The reason behind adopting a different approach from Dittmann et al. [22] regarding the taxonomies of *Component*, *Function* and *Failure*, is the need for facilitating the calculation of similarities among problems. A *Component* taxonomy with more levels (i.e., with more than one level) would create an overlap of information with the taxonomy of *Function*. That overlap could affect the similarity calculation and the retrieval of similar cases within a CBR system. For example, a *Component* of a hypothetical subtype 'component > process > stamping' modifies the shape of a metal sheet. This *Component* is linked to a *Problem* that has an associated *Function* of subtype 'function > modify > modify material > modify shape', that does not add any additional information. Opposite, both parameters with same information will reduce the probabilities that the CBR system is able to associated this query with applicable cases from a *Component* of a hypothetical subtype 'component > process > extrusion', which is also related to the function 'function > modify > modify material > modify shape', and which could bring valuable solutions to the user. The use of a taxonomy with the class *Failure* is very similar. In PFMEA, failures relate to unfulfilled functions (e.g.: Function 1 = 'metal sheet is drawn 20 mm'/Failure 1 = 'metal sheet is not drawn'/Failure 2 = 'metal sheet is drawn less than 20 mm'/Failure 3 = 'metal sheet is drawn more than 20 mm'). Therefore, a taxonomy would not add relevant information, and the degree of fulfilment can be better represented with a percentage value.

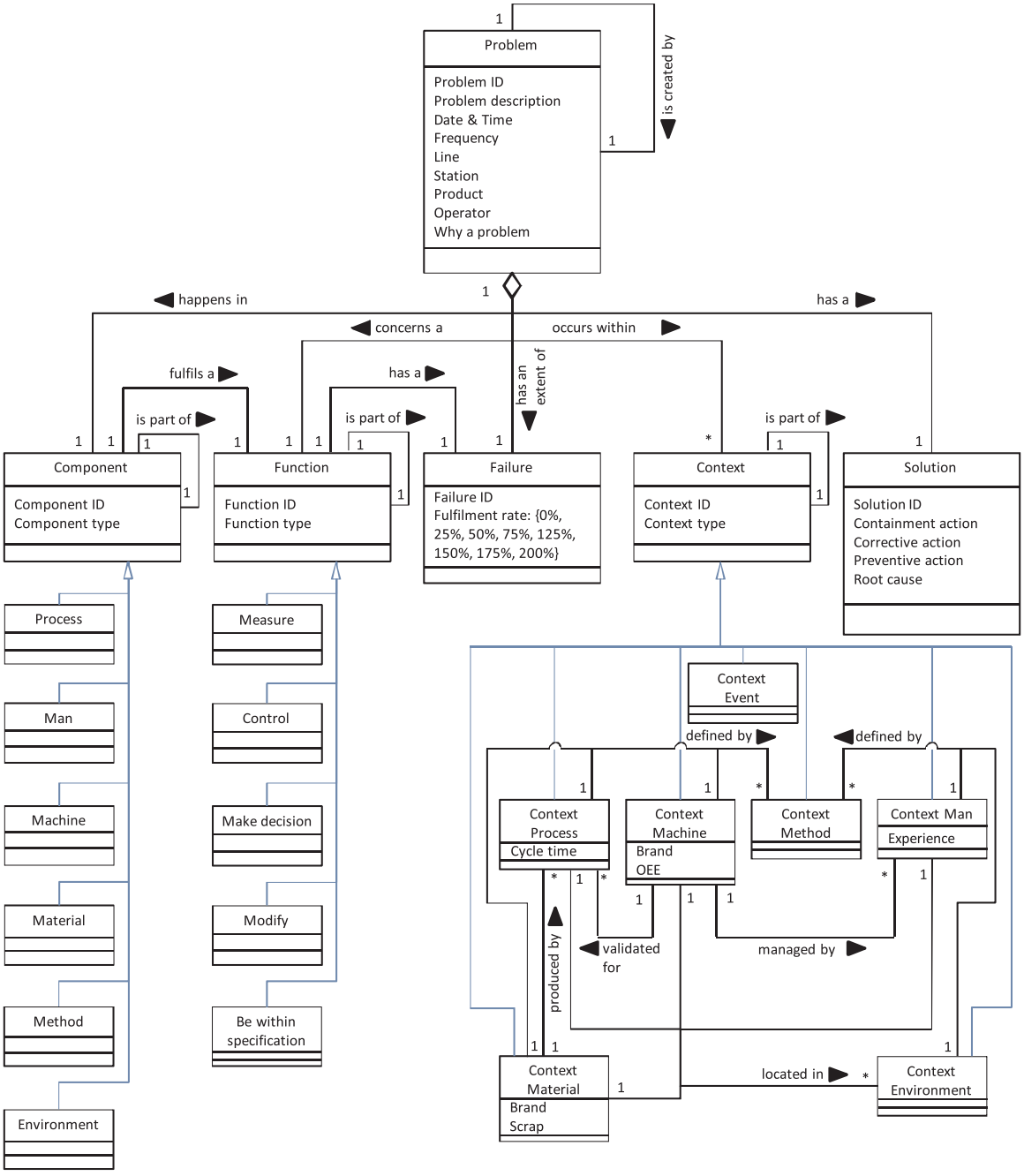

Fig. 2. Main components of the proposed ontology in a lightweight UML representation.

The proposed model should be aligned with the definition of problem according to MPS. Opposite to PFMEA, it considers a problem as a specific and unique failure happening in a specific and unique function that belongs to a specific and unique component. Therefore, the class *Problem* was added to the ontology to provide the link to a unique trio of *Component*, *Function* and *Failure*. This approach reflects one of the main differences between PFMEA and MPS. While in PFMEA each failure mode can belong to different functions that at the same time can belong to different components, in an MPS situation an employee faces a specific and unique problem (i.e., failure) that comprises a single combination of component, function and failure. For example, in a typical PFMEA approach a component pneumatic cylinder can have the functions of being tight, moving forward, and moving backward. The first function may fail by leaking between chambers, or by leaking to the outside. The next two functions may fail by moving too fast, too slow, or not moving at all.

Usually, a top-level general problem is created by a lower-level more-specific problem. This relationship between a problem and its lower-level creator is represented by the relationship '*is created by*' in the class *Problem*. The class *Problem* has also different attributes to represent the basic description of a manufacturing problem, as mentioned in Section 3.1. Each attribute has an associated taxonomy defining possible values. This approach helps calculate the similarity among problems (e.g., problems originated in the same production line should have a higher similarity index than ones originated in other lines). The definition of a problem, through the classes *Component*, *Function* and *Failure*, helps classify a problem in a generic way while the attributes of the class *Problem* define its details.

Although the mentioned trio of *Component*, *Function* and *Failure* defines a specific problem, the exact same trio can be found in different

manufacturing scenarios. For instance, a problem with an electrical relay, which does not allow the flow of current when it is activated, is linked to a component of type 'machine' with the function 'modify > modify flow > increase flow' that fails by not fulfilling at all its function (i.e., its *Fulfilment rate* is 0%). However, such a description of a problem could refer to a small relay in an electronic board of a PLC (Programmable Logic Controller) controlling micro-amperes or to a big device in the control unit of a power station controlling mega-amperes. To address this issue, the class *Context* was defined to add information related to the environment of the problem. The work of Chungoora et al. [18], as mentioned in Section 2.1, was taken as the reference for the ontological definition of *Context*. In particular it was taken the classes *Method* and *Process*, and three subclasses that specialize the class *Resource*: *Man*, *Material*, and *Machine*. In order to match the six components of the PFMEA, the class *Environment* was added to the model. The directions of the relationships among them have been adjusted to facilitate the search for information in the PLM system. The subclass *Event* has been included to add valuable information related to special events that may occur before a problem is detected (e.g., change over, change of the production shift, or restart of the production after a weekend). The *Event* subclass is relevant in the problem similarity calculation. It was realized by observing how experts conduct an MPS process. Experts usually start the investigation by requesting information about any relevant event that happened just before the problem occurred. The class *Context* has seven subclasses, as the class *Component*, and has an associated taxonomy represented through a relationship type "is part of" pointing to itself. The subclasses of *Context* have many different types of associated attributes, which are used to specify every type of technical information of the context (e.g., pressures, temperatures, and dimensions). These attributes are used in the configuration of the PLM system to store PPR information explicitly.

Finally, the class *Solution* contains information about the solution to the manufacturing problem. This class contains information related to the containment, corrective, and preventive actions applied to solve the problem. This information is not used in the similarity calculation.

3.3. Manufacturing problem solving system architecture

From an IT perspective, the proposed process model is based on the mentioned SEASALT multi-agent architecture [13]. Fig. 3 shows its application to the proposed MPS system. The proposed MPS system architecture supports the deployment of the different agents across the different manufacturing plants of a company (hereafter referred to as locations). Within each location, it supports the deployment across the areas with different manufacturing processes or production units. In this way, each agent hosted in a specific production unit of a specific location will be able to communicate and interchange information with all the other agents hosted in different production units and locations through the company's intranet.

The SEASALT architecture has been simplified, taking only those parts that are relevant to this research. The simplification is based on the work of Mikos et al. [23], who propose three different types of agents to manage all needed functions around PFMEA knowledge interchange. The three agent types correspond to the following ones in the SEASALT architecture:

- Individualized Knowledge Agent: it is responsible for capturing and showing information to the user. In this work, the tasks of this agent, defined in [13] and [23], are extended to include the collection of context information from the PLM system to complete the definition of a problem. There are as many agents of this type as users of the MPS system.
- Topic Agent: it is responsible for calculating similarities through a CBR application and proposing the best solution out of its specific case base. There are as many agents of this type as production units, and each of them is hosted in the central device of its corresponding production unit.
- Coordination Agent: it is responsible for the communication coordination among agents and the selection of the best solution among the ones proposed by the Topic Agents. There are two types of coordination agents. A global coordination agent performs a global coordination role in the architecture among all connected locations. A plant coordination agent performs the coordination of a single location. It takes as input a Knowledge Map containing the list of the Topic Agents working in its location. There is a single global coordinator and as many plant coordinators as locations. The global coordinator is hosted in the central server of the company while the plant coordinators are hosted in the servers of each location.

The Knowledge Representation module corresponds with the proposed ontological model presented in the previous section.

The following section presents the application of these models in a proof of concept MPS system prototype implemented in the multinational company Exide Technologies with the aim of testing and validating its functionality and capabilities.

4. Manufacturing problem solving system prototype

4.1. Development of the proof of concept prototype

The proposed models were used to develop an MPS system prototype. Two software applications were selected for the development: Aras Innovator (partially open source) as PLM software and myCBR (open source) as CBR system. Both systems were customized to support the proposed MPS Process Model and MPS ontology. An initial standalone prototype was created to perform a preliminary evaluation [33].

The MPS system prototype was programmed in Java. The three different agent types (Individualized Knowledge, Topic, and Coordination) were programmed to be launched in the JADE run-time environment. myCBR, which is an open source piece of software developed in Java, was integrated within the Topic Agent code. The communication with the PLM is done via HTTP between the Individualized Knowledge Agent and Aras server, and is based on AML (Aras Markup Language) messages. AML is the XML (eXtensible Markup Language) dialect and language that drives the Aras Innovator server.

Fig. 4 shows the Graphical User Interface (GUI) of the developed prototype system. The GUI shows a Problem Solving Sheet (PSS) that guides graphically the user through the eight steps of the 8D Method. PSS is very common in the industry. Therefore, typical users are familiar with it. Adopting such a GUI should facilitate the intereaction with the users.

4.2. Configuration and initial case base of the MPS system prototype

The MPS system prototype was implemented in two manufacturing plants, one located in Germany (Plant A) and one located in Spain (Plant B). Wet Filling was the common manufacturing process selected for the prototype testing. Wet Filling is a process for producing positive plates for industrial batteries.

Staff from production and engineering in the German plant was interviewed to identify the characteristics of the Wet Filling industrial reality that make products, machines, processes, workers, and environments different from one another. These differences are the key to distinguish a problem from another in the similarity calculation to be executed by the CBR system. The information gathered in the interviews was: key elements, their relationships, and relevant attributes.

For the calculation of similarities the standard set of formulas available in the selected software myCBR was used. myCBR provides a GUI for modelling various kinds of attribute-specific similarity measures and for evaluating the resulting retrieval quality [34]. It follows the local-global approach, which divides the similarity definition into a set of local similarity measures for each attribute, a set of attribute weights, and a global similarity measure for calculating the final

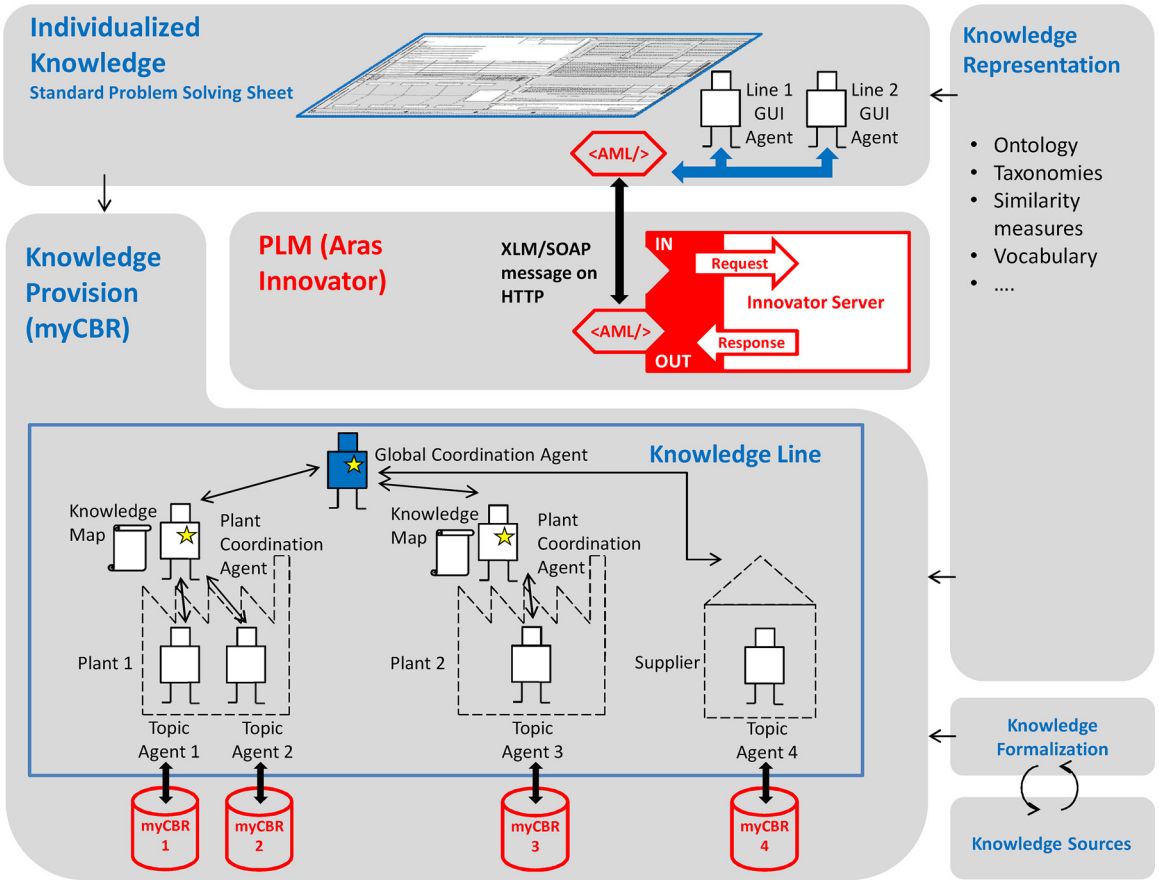

Fig. 3. Proposed MPS system architecture based on SEASALT.

similarity value. This means, for an attribute-value based case representation consisting of n attributes, the similarity between a query q and a case c is calculated as follows (Eq. (1)) [35]:

$$\text{Sim}\ (q, c) \sum_{i=1}^{n} w_i \cdot \text{sim}_i(q_i, c_i) \tag{1}$$

For the configuration of local similarities, it is distinguished between numerical attributes (representing technical specifications of products, processes or environment) and the attributes with an associated taxonomy. For the first ones, the similarity computation is based on a mapping between the distance of the query q and the case c (Eq. (2)) [35]:

$$\text{sim}_i(q_i,c_i) = f(d(q_i,c_i)) = 1 - |c_i - q_i|/|\text{Max allowed value} - \text{Min allowed value}| \tag{2}$$

For the attributes with an associated taxonomy, the similarity calculation is based on the relative position of query and case values within the taxonomy. For this purpose, the defined taxonomies are enhanced in myCBR with position factors. Every inner node K_i of the taxonomy is associated to a position factor $S_i \in [0.1]$, which is defined based on experience and refined through trial and error. This factor holds the following condition: if $K_1 > K_2$ then $S_1 \leq S_2$. The deeper the nodes are located in the hierarchy, the larger the similarity value can become. The similarity between two objects is defined as (Eq. (3)) [36]:

$$\text{Sim}\ (K_1, K_2) \begin{cases} 1 & if K1 = K2 \\ S\langle K1, K2\rangle & otherwise \end{cases} \tag{3}$$

where $S < K_1,K_2 >$ denotes the position factor assigned to the node $< K_1,K_2 >$, i.e., the nearest common predecessor of K_1 and K_2.

For the configuration of weights for each local similarity result, it was initially decided that the weight of the attributes describing the problem (i.e., component, function, failure, and problem) was double the weight of the attributes describing the context (i.e., man, machine, material, environment, method and event). The aim behind this decision was that the problems with a similar definition, based on the taxonomies, would be considered similar by the CBR system. And then the context definition would be used as complement to distinguish the closest cases to the query, with both problem and context definition similar.

One of the aims of this work was the easy reutilization of potential problems recorded in PFMEA analyses. In this case, the PFMEA of the Wet Filling process was taken as the initial source of cases for the MPS prototype case base. The initial set of cases was extended with problems collected during one week in the Wet Filling shop floor of the Plant A. For the evaluation phase of the prototype a total of 72 different chains of problems (from visible problem to root cause) were recorded, which corresponds with 226 individual problems (each chain of problems comprises several individual problems).

4.3. Case Studies and Validation of the MPS system prototype

For the first validation step of the prototype system, the Wet Filling production area in Plant A was selected (Case A). Case A represents the lowest level of complexity for the system, since the cases were all collected in this process and plant, as described in the Section 4.2. The Wet Filling production area in Plant B was selected for the second validation step of the prototype system (Case B). Case B represents a higher level of complexity for the system, since it occurs in a different manufacturing context (the plant) than where the knowledge was collected.

The types of problems in both plants can be quite different (e.g., machines can be different, some materials are bought from local suppliers, and personnel have different levels of training and experience).

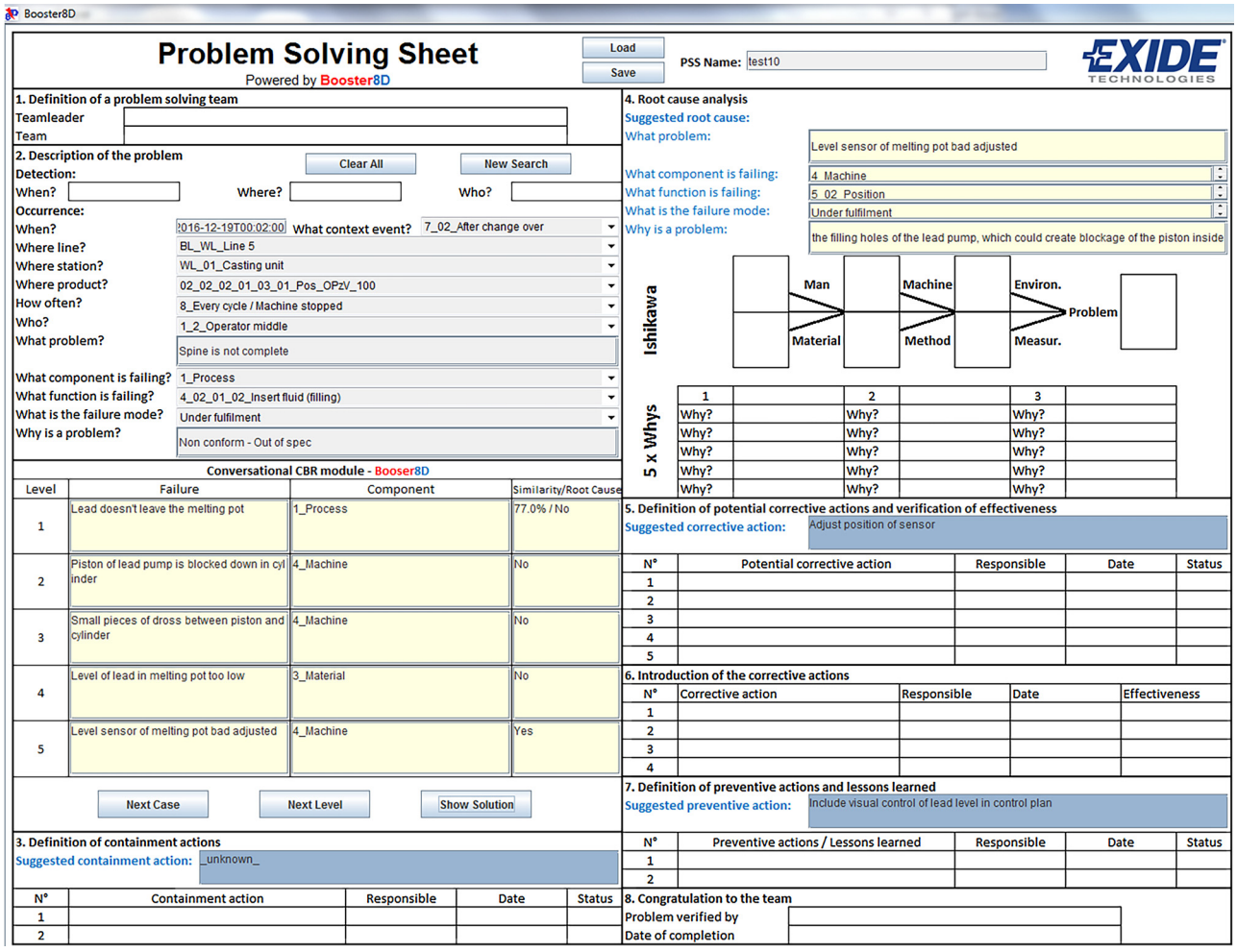

Fig. 4. Problem Solving Sheet GUI of the prototype MPS system.

Finally, for the third validation step of the prototype system, the Casting production area in the Plant A was selected (Case C). This Case C represents the highest level of complexity for the system, because the case base did not contain any kind of problem from this area and the PLM was not set up with data about this area either. In the three cases the corresponding group leaders were trained and used the system to solve ten problems that arose during their shifts. Queries, results, and real solutions were all recorded for a detailed analysis and evaluation afterwards.

When the user introduces the initial definition of a problem through the prototype GUI (Fig. 4), the individualized knowledge agent hosting the interface uses this information to send queries to the PLM system and collect all the associated context information. Initially, and based on the user input information related to line, station, product, and date of the problem, this agent retrieves the IDs (IDentification numbers) of the PLM items that were related to machine and product involved in the problem under analysis and that were active in the PLM in the given date. With these two IDs, the agent can extract the ID of the rest of related items (i.e., man, process and environment). When the items' ID are known, the individualized knowledge agent is able to place queries directly on each selected item and extract all the information linked to it (e.g., supplier of the machine, experience of the operator, or maximum casting temperature). Fig. 5 shows an example of query and answer based on the language AML of Aras. Because the basic communication channel among agents defined in JADE are Java Strings, the individualized knowledge agent puts together all the collected parameters into a single string (i.e., the user query data plus the PLM extracted context data). This is done by using a tag language similar to

Query to PLM

```
<AML>
  <Item type="Part" action="get">
    <name>02_02_02_01_03_01_Pos_OPzV_100</name>
  </Item>
</AML>
```

Answer from PLM

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/>
  <SOAP-ENV:Body>
    <Result>
      <Item type="Part" typeID="4F1AC04A2B484F3...." id="3F3829DC968D421....">
        <created_on>2016-12-10T21:53:19</created_on>
        <current_state keyed_name="Released">42BB3B183A77...</current_state>
        <brand keyed_name="Internal" type="Manufacturer">6E5264B47A15....</brand>
        <component_type>3_02_02_Assembly</component_type>
        ....
      </Item>
    </Result>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Fig. 5. Example of query and PLM answer based on AML.

HTML (Fig. 6). Then the individualized knowledge agent sends the created string to the coordination agent, which distributes it simultaneously among all the existing topic agents. Each topic agent extracts the query parameters from the string and uses them to retrieve solutions from the CBR application hosted in the agent. The best proposed solutions, together with their corresponding similarity percentage related

Query

```
<Query>
        User query
        <When>2016-12-19T00:02:00</When>
        <Whereline>BL_WL_Line 5</Whereline>
        <Wherestation>WL_01_Casting unit</Wherestation>
        <Whereproduct>02_02_02_01_03_01_Pos_OPzV_100</Whereproduct>
        <Howoften>8_Every cycle / Machine stopped</Howoften>
        <Who>1_2_Operator middle</Who>
        <Whatproblem>Spine is not complete</Whatproblem>
        <Component>1_Process</Component>
        <Function>4_02_01_02_Insert fluid (filling) </Function>
        <Failure>Under fulfilment </Failure>
        <Whyproblem>Non conform - Out of spec</Whyproblem>
        <Context_Event>7_02_After change over</Context_Event>
        PLM contribution to query
        <2_1_Context Process>1_02_04_01_02_01_Die Casting</2_1_Context Process>
        <2_Casting Temperature Max>…</2_Casting Temperature Max>
        …
        <2_2_Context Man>2_01_Operator</2_2_Context Man>
        <2_Experience>534.0</2_Experience>
        …
        <2_3_Context Material>3_02_02_Assembly</2_3_Context Material>
        <2_Material Brand>Internal</2_Material Brand>
        …
        <2_4_Context Machine>4_02_04_Modifying element</2_4_Context Machine>
        <2_Machine Brand>Hadi</2_Machine Brand>
        …
</Query>
```

Solution

```
<Case>
      <Case1>
            <Similarity>0.8157281042128605</Similarity>
            <Level1>
                  <Containment>_unknown_</Containment>
                  <Corrective>With a long drill bit re-drill the lead pump
                  hole</Corrective>
                  <Preventive>_unknown_</Preventive>
                  <What_sol>Flow of lead reaching the mould is very small because
                  the exit hole of the pump is partially blocked</What_sol>
                  <Component_sol>1_Process</Component_sol>
                  <Function_sol>4_02_01_02_Insert fluid (filling)</Function_sol>
                  <Failure_sol>Under fulfilment</Failure_sol>
                  <Why_sol>Lead gets too slow in the mould and it gets too
                  cold</Why_sol>
                  <Root_case>Yes</Root_case>
            </Level1>
      </Case1>
      <Case2>
            <Similarity>0.7748190133037695</Similarity>
            <Level1>
                  …
            </Level1>
            <Level2>
                  …
            </Level2>
      </Case2>
      <Case3>
      .......
</Case>
```

Fig. 6. Example of query and solution expressed in tag language.

to the query, are built again into a string using the same tag language. The created string is then sent to the coordination agent. This coordination agent consolidates all the answers from the different topic agents and selects the proposals with the highest similarity values. It puts all top proposals together in a single string (Fig. 6), and sends it to the individualized knowledge agent, which extracts the information and shows it to the user through the GUI (Fig. 4).

The Table 1 shows the results extracted from the three cases. The table is divided horizontally in three blocks corresponding to Cases A, B, and C. The table is divided vertically into three blocks, which correspond to the three consecutives improvements done with the similarity parameters of the CBR system. Finally, the table shows, for each problem, the similarity percentage of the proposal given by the system to each user query and the codes to interpret the result.

- −1 = the MPS system proposed some solutions but none of them were useful because the case base did not contain useful cases. This represents a failure of the MPS system due to lack of knowledge.
- 0 = the MPS system proposed some solutions but none of them were useful. Nevertheless, in the case base, there was one suitable case at least. This represents a failure of the MPS system due to a failure in the similarity calculation.
- 1 = the MPS system proposed at least one solution that could be directly used without any major adaptation (i.e., the proposal comes from an identical or very similar context).
- 2 = the MPS system proposed at least one solution that could be used after an adaptation process done by the user (i.e., the proposal comes from a different context).

Initially the system was configured to give to the user just three proposals of solution, because it was considered that if the three most similar proposals from the agents do not match, the better would be that the user proceeds to reformulate the query. The results from the first trial are interpreted as follow:

- 80% of the proposals in Case A were suitable, having most of them code 1 (i.e., directly useful without adaptation). This result is obvious, since the case base was populated with problems coming from this area.
- Only 60% of the proposals in Case B were suitable, having most of them code 2 (i.e., adaptation was needed). A high level of adaptation is considered normal since the case base was populated with problems coming from the Plant A, which implies a different manufacturing context. Nevertheless, a 60% success rate is considered too low.
- Only 10% of the proposals in Case C were suitable. This can be explained because the PLM system was not populated with information from the area of this case. Therefore, most of the parameters to calculate similarity were empty.

The results from the first trial were not considered satisfactory. A deeper analysis of the CBR similarity configuration was considered necessary to optimize the success rate of the system. It was realized that the selected strategy of weights associated to each attribute (see Section 4.2) was inadequate, since the number of attributes associated to the context, which were extracted from the PLM system, was much higher than the number of attributes associated to the user definition of the problem. Therefore, even though the problem related attributes had higher individual weights, all the context related attributes together had a much higher impact into the similarity calculation. This made the system perform well only when the problem under analysis had the same context as any of the problems in the case base. To solve this issue, the approach to weights was changed, grouping attributes into blocks and giving them a total weight as a unit. The flexibility and visualization of results in myCBR [34] helped in the identification and correction of errors during the development of the system. The new configuration was:

- 40% weight of total similarity result: core problem description (i.e., component, function, and failure). This gives a very high relevance to these three parameters, which work with universal taxonomies regardless of the context type, ensuring the retrieval of useful cases from different contexts.
- 40% weight of total similarity result: core context description (i.e., process, man, machine, material, environment, method, and event) and the parameters to define ‘who’ and ‘how often’. As with the previous parameters, these nine parameters have also universal taxonomies, ensuring the retrieval of useful cases from different contexts. In this case, the 40% weight has to be split among nine units, instead of three, having less impact at the individual level.
- 10% weight of total similarity result: line, station, and product. These parameters have no universal taxonomy (i.e., they are specific to each production unit). Thus, it is possible that the CBR system, linked to agents working in different areas from the one where the user is making the query, will not recognize them. Therefore, they receive less weight.
- 10% weight of total similarity result distributed homogeneously through the whole set of context technical parameters extracted

Table 1
Results of cases A, B and C with different similarity configurations.

	Initial similarity configuration 3 proposals		Improved similarity configuration 3 proposals		Improved similarity configuration 10 proposals	
	Code	%	Code	%	Code	%
Case A: German Plant - Wet Filling						
A01	1	86,2	1	85,2	1	85,2
A02	1	79,5	1	83,8	1	83,8
A03	1	85,2	1	82,2	1	82,2
A04	1	86,2	1	85,2	1	85,2
A05	0	0	0	0	1	77,1
A06	-1	0	-1	0	-1	0
A07	1	79,8	1	70,1	1	70,1
A08	1	76,2	0	0	1	67,2
A09	2	88,6	2	84	2	84
A10	2	64,5	1	72,8	1	72,8
Summary of Results Case A						
A: NOK	2	20%	3	30%	1	10%
A: OK	8	80%	7	70%	9	90%
A: -1	1	10%	1	10%	1	10%
A: 0	1	10%	2	20%	0	0%
A: 1	6	60%	6	60%	8	80%
A: 2	2	20%	1	10%	1	10%
Case B: Spanish Plant - Wet Filling						
B01	1	77,8	1	77,9	1	77,9
B02	2	68,4	2	65,7	2	65,7
B03	1	80,5	1	84,2	1	84,2
B04	2	71,8	2	73,4	2	73,4
B05	0	0	0	0	2	76
B06	0	0	1	83,8	1	83,8
B07	2	81,6	1	82,8	1	82,8
B08	-1	0	-1	0	-1	0
B09	-1	0	-1	0	-1	0
B10	2	80,5	2	81,2	2	81,2
Summary of Results Case B						
B: NOK	4	40%	3	30%	2	20%
B: OK	6	60%	7	70%	8	80%
B: -1	2	20%	2	20%	2	20%
B: 0	2	20%	1	10%	0	0%
B: 1	2	20%	4	40%	4	40%
B: 2	4	40%	3	30%	4	40%
Case C: German Plant - Grid Casting						
C01	0	0	0	0	0	0
C02	-1	0	-1	0	-1	0
C03	0	0	0	0	0	0
C04	2	84,1	2	75,1	2	75,1
C05	0	0	0	0	0	0
C06	-1	0	-1	0	-1	0
C07	0	0	2	68,1	2	68,1
C08	-1	0	-1	0	-1	0
C09	0	0	0	0	0	0
C10	-1	0	-1	0	-1	0
Summary of Results Case C						
C: NOK	9	90%	8	80%	8	80%
C: OK	1	10%	2	20%	2	20%
C: -1	4	40%	4	40%	4	40%
C: 0	5	50%	4	40%	4	40%
C: 1	0	0%	0	0%	0	0%
C: 2	1	10%	2	20%	2	20%

from the PLM system (e.g., pressure, temperature, dimensions, etc.).

The same queries were introduced into the system using this second similarity configuration setup. The results are shown in the second vertical block of Table 1. Cells filled in green color show queries with improved results while those filled in red color show queries with worse results:

- 70% of the proposals in Case A were now suitable. This represented a setback when compared to the previous trial configuration. With this configuration, the core description of the problem has more relevance. Thus, the system is able to find many more similar cases from different contexts and the probability of finding the proper one within the three proposals delivered by the system is lower.
- 70% of the proposals in Case B were now suitable, which means a

10% improvement. The proposals demanding adaptation decreased as well.
- 20% of the proposals in Case C were now suitable, which means a 10% improvement.

The similarity calculation done by the system with this second configuration was considered right. However, the increase of relevancy given to the problem core description has generated that many proposal given by the system are now linked to a unique context. This effect is especially visible when the most similar cases proposed by the system were initially collected from a PFMEA document. It should be remembered that the goal of PFMEA is to looks for all possible failure modes linked to each component in a specific process (i.e. context). Since the system in these two initial configurations was defined to give to the user only the three most similar proposals, quite probably, it will only give information about a problem from a single context. In the case that this context is not matching with the one of the user, the system will not allow other possibilities to show up. For this issue, the system was reconfigured to show the top ten out of all the agent proposals. The new results are shown in the third vertical block of Table 1. Cells filled in green color show queries with improved results related the first configuration, while those filled in red color show queries with worse results:

- 90% of the proposals in Case A were now suitable, which represented an improvement of 10% in comparison to the first trial and of 20% to the second one.
- 80% of the proposals in Case B were now suitable, which represented an improvement of 20% in comparison to the first trial and of 10% to the second one.
- 20% of the proposals in Case C were now suitable, which represented an improvement of 10% in comparison to the first trial and no improvement to the second one.

As it was mentioned in Section 3.2, the proposed model should support any kind of manufacturing process and any location where a manufacturing process takes place. Therefore, there are no theoretical limitations when it comes to its applicability within the manufacturing environment. Nevertheless, the degree of utility of the proposed model will be strongly limited by the availability of useful cases stored in the case base. In terms of implementation, the generality of the approach is improved when: the context dependent taxonomies of the parameters used to define the problem (i.e., line, station, product and operator) is extended with data of any new location, the generic data in the taxonomies of the classes *Function* and *Context* is extended with enough detail to support any new manufacturing scenario, and finally the PLM system contains any new context related information.

5. Conclusions

The results obtained with the developed proof-of-concept Manufacturing Problem Solving (MPS) system demonstrated the feasibility of the approach adopted in this work. Its goal is to assist shop floor manufacturing employees, such as operators or quality inspectors, in the execution of MPS processes, which are characterized by knowledge intensive activities heavily based on experiences.

The main contribution of this work is its innovative production-oriented approach to MPS by combining classic MPS methods with CBR on an agent-based distributed architecture, and with a PLM system. This novel approach had not been proposed in the reviewed literature. The proposed approach integrates 8D, PFMEA (Process Failure Mode and Effect Analysis), Case-Based Reasoning (CBR), and Product Lifecycle Management (PLM). 8D is chosen as the MPS method and PFMEA as a preventive technique to create an initial set of potential manufacturing knowledge base. A CBR application, on an agent-based distributed architecture, is chosen as the artificial intelligence tool for searching for similar manufacturing problem cases. Finally, a PLM application contains extended context information to enrich the similarity calculation within the CBR application.

The application of the SEASALT architecture (Shared Experience using an Agent-based System Architecture LayouT) [13] to an industrial environment of production can also be considered as a contribution.

Another relevant contribution of this work is the ontological approach. It combines and extends existing ontologies from the PFMEA [22] and manufacturing [18] environments to create a new ontology able to represent manufacturing problems linked to production lines. The proposed ontology supports at the same time the reuse of PFMEA analysis results.

Future research and development work can be summarized as follows:

- The extension of the developed system with a connection to a Manufacturing Execution System (MES) to incorporate automatically context event information, such as change overs, starts of shifts, or change of operators at the line.
- The development of the adaptation container in myCBR to have automatic adaptation of the solutions proposed by the system to the specific context of each query.
- The systematic record of manufacturing problems in the developed system would encourage the development of applications to perform a statistical analysis of the data (e.g., number of failures related to specific components, number of failures where a specific supplier is involved, number of failure of a type of machine component, etc.) leading to analyzing the use of machine learning techniques to make predictions and assist in decision making tasks.
- The extension of the use of the SEASALT architecture to include the development of the Knowledge Source and the Knowledge Formalization modules to extract knowledge from the PFMEA case base automatically.
- The development of a multi-language dictionary to support the parallel use of the system by users with different native languages.
- The use of semantic methods to collect information directly from free text describing a manufacturing problem.

References

[1] VDA. Qualitätsmanagement in der Automobilindustrie – Qualitätsmanagement-Methoden Assessments. Verband der Autoindustrie (VDA); 2015.
[2] Bhuiyan N, Baghel A. An overview of continuous improvement: from the past to the present. Manag Decis 2005;43(5):761–71.
[3] De Mast J, Lokkerbol J. An analysis of the Six Sigma DMAIC method from the perspective of problem solving. Int J Prod Econ 2012;139:604–14.
[4] Rother M. Toyota kata: managing people for improvement, adaptiveness, and superior results. McGraw-Hill Professional; 2010.
[5] Shainin RD. Statistical engineering six decades of improved process and systems performance. Qual Eng 2012;24(2):171–83.
[6] Womack JP, Jones DT, Roos D. The machine that changed the world – the story of lean production. Productivity Press; 1991.
[7] Dennies DP. The organization of a failure investigation. Methodology 2002;2(3):11–41.
[8] Riesenberger CA, Sousa SD. The 8D methodology: an effective way to reduce recurrence of customer complaints. World Congr Eng 2010;3.
[9] Bhote KR, Bhote AK. World class quality – using design of experiments to make it happen. AMACOM – American Management Association; 1999.
[10] Liu DR, Ke CK. Knowledge support for problem-solving in a production process: a hybrid of knowledge discovery and case-based reasoning. Expert Syst Appl 2007;33:147–61.
[11] Lundgren M, Hedlind M, Kjellberg T. Model driven manufacturing process design and managing quality. Procedia CIRP 2016;50:299–304.
[12] Richter MM, Weber RO. Case-based reasoning: a textbook. Heidelberg: Springer; 2013.
[13] Bach K. Knowledge acquisition for case-based reasoning systems. Ph.D. thesis. University of Hildesheim; 2012.
[14] Liu W, Zeng Y, Maletz M, Brisson D. Product lifecycle management: a review.

Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2009.
[15] Stark J. Product lifecycle management. Springer International Publishing; 2015.
[16] Chandrasekaran B, Josephson JR, Benjamins VR. What are ontologies, and why do we need them? IEEE Intell Syst 1999;14(1):20–6.
[17] Lemaignan S, Siadat A, Dantan JY, Semenenko A. MASON: a proposal for an ontology of manufacturing domain. IEEE workshop on distributed intelligent systems: collective intelligence and its applications, DIS 2006:195–200.
[18] Chungoora N, Young R, Gunendran G, Palmer C, Usman Z, Anjum N, et al. A model-driven ontology approach for manufacturing system interoperability and knowledge sharing. Comput Ind 2013;64(4):392–401.
[19] Kamsu-Foguem B, Couderta T, Bélera C, Genestea L. Knowledge formalization in experience feedback processes: an ontology-based approach. Comput Ind 2008;59(7):694–710.
[20] Scippacercola F, Pietrantuono R, Russo S, Silva NP. SysML-based and prolog-supported FMEA. In Software Reliability Engineering Workshops (ISSREW), 2015 IEEE International Symposium 2015:174–81.
[21] Ebrahimipour V, Rezaie K, Shokravi S. An ontology approach to support FMEA studies. Expert Syst Appl 2010;37(1):671–7.
[22] Dittmann L, Rademacher T, Zelewski S. Performing FMEA using ontologies. 18th international workshop on qualitative reasoning 2004:209–16.
[23] Mikos WL, Ferreira JCE, Botura PEA, Freitas LS. A system for distributed sharing and reuse of design and manufacturing knowledge in the PFMEA domain using a description logics-based ontology. J Manuf Syst 2011;30:133–43.
[24] Bertin A, Noyes D, Sanchez P. Lessons learned system integration into a PLM tool. International conference on product lifecycle management 2010:577–89.
[25] Bertin A, Noyes D, Clermont P. Problem solving methods as Lessons Learned System instrumentation into a PLM tool. 14th IFAC symposium on information control problems in manufacturing 2012.
[26] Wuest T, Weimer D, Irgens C, Thoben KD. Machine learning in manufacturing: advantages, challenges, and applications. Prod Manuf Res 2016;4(1):23–45.
[27] Susto GA, Schirru A, Pampuri S, McLoone S, Beghi A. Machine learning for predictive maintenance: a multiple classifier approach. IEEE Trans Ind Inf 2015;11(3):812–20.
[28] Plaza E, McGinty L. Distributed case-based reasoning. Knowl Eng Rev 2005;20(3):261–5.
[29] Althoff KD, Bach K, Deutsch JO, Hanft A, Mänz J, Müller T, et al. Collaborative multi-expert-systems. Proceedings of the 16th UK workshop on case-based reasoning 2012:13.
[30] Reuss P, Althoff KD, Hundt A, Henkel W, Pfeiffer M. Multi-Agent case-based diagnosis in the aircraft domain. Workshop proceedings – 23rd international conference on case-based reasoning 2015:43–52.
[31] Kepner CH, Tregoe BB. The new rational manager – an updated edition for a New World. Princeton Research Press; 2008.
[32] Becerra-Fernandez I, Aha DW. Case-based problem solving for knowledge management systems. 12th International FLAIRS Conference 1999.
[33] Camarillo A, Ríos J, Althoff KD. CBR and PLM applied to diagnosis and technical support during problem solving in the continuous improvement process of manufacturing plants. Procedia Manuf 2017;13:987–94.
[34] Roth-Berghofer T, Sauer C, Garcia JAR, Bach K, Althoff KD, Agudo BD. Building case-based reasoning applications with myCBR and COLIBRI studio. Proceedings of the UKCBR 2012.
[35] Stahl A, Roth-Berghofer TR. Rapid prototyping of CBR applications with the open source tool myCBR. Adv Case-Based Reason 2008:615–29.
[36] Bergmann R. On the use of taxonomies for representing case features and local similarity measures. Proceedings of the 6th German Workshop on case-based reasoning 1998.