A family of case studies on business process mining using MARBLE

Ricardo Pérez-Castillo*, José A. Cruz-Lemus, Ignacio García-Rodríguez de Guzmán, Mario Piattini

Instituto de Tecnologías y Sistemas de Información at University of Castilla-La Mancha, Spain

ARTICLE INFO

ABSTRACT

Business processes, most of which are automated by information systems, have become a key asset in organizations. Unfortunately, uncontrolled maintenance implies that information systems age overtime until they need to be modernized. During software modernization, ageing systems cannot be entirely discarded because they gradually embed meaningful business knowledge, which is not present in any other artifact. This paper presents a technique for recovering business processes from legacy systems in order to preserve that knowledge. The technique statically analyzes source code and generates a code model, which is later transformed by pattern matching into a business process model. This technique has been validated over a two-year period in several industrial modernization projects. This paper reports the results of a family of case studies that were performed to empirically validate the technique using analysis and meta-analysis techniques. The family of case studies demonstrates that the technique is feasible in terms of effectiveness and efficiency.

© 2012 Elsevier Inc. All rights reserved.

1. Introduction

Business processes are increasingly becoming an essential asset for organizations since they create value for customers and reflect all operations of an organization (Chang, 2006). Business processes depict a set of related activities that are performed in an organization and that together realize a business goal (Weske, 2007). These descriptions provide a means to map business objectives regarding how to best carry out operations. Organizations adopt business process management through their enterprise information systems. Business processes therefore also assist at the beginning of the development of information systems to automate all those activities needed to achieve the business objectives (Heuvel, 2006).

However, enterprise information systems are not static entities, since all software ages as a result of uncontrolled maintenance over time (Visaggio, 2001). The immediate effect of software ageing is that information systems become *Legacy* Information Systems (LIS) and must therefore be replaced with a new, improved system. This activity is known as *software modernization*. Unfortunately, when organizations modernize their legacy information systems, the organizations' business processes do not reflect all changes that have occurred during the maintenance of their systems. Indeed,

very often the business process documentation is neither updated nor documented at all (Lehman, 1984) Thereby, the business process models considered by the organization are not aligned with the actual business process that is executed through legacy information systems (Heuvel, 2006). Owing to the alignment problem, a legacy information system cannot be entirely discarded during its modernization since it might contain a considerable amount of latent meaningful business knowledge (Koskinen, 2004). This knowledge is embedded in the system as a consequence of modifications over time and it might not, therefore, be present anywhere else. As a result, all embedded business processes must be explicitly recovered in order to preserve this meaningful asset in the modernized information systems (Paradauskas and Laurikaitis, 2006). The recovered business processes can then be used by organizations in two ways: (i) to provide business experts with a better understanding of the real, current operation of the organization; and (ii) if necessary, to develop the new improved system in order to mitigate the effects caused by the software ageing. The evolved system will thus support the current business processes and will also improve the ROI (*Return Of Investment*) of the legacy system, since it extends its lifespan.

This paper deals with the problem of business process mining from legacy information systems in order to address the challenge of business knowledge preservation. The paper provides a semi-automatic technique for recovering business processes from legacy information systems. A supporting tool is provided to automate and facilitate the adoption of the technique. The proposal consists of a reverse engineering technique that follows the model-driven

development principles. The technique therefore depicts a set of models and the transformations between them to obtain a set of preliminary business processes. The technique is aided by the manual post-intervention of business experts who refine the business processes obtained (e.g., by adding manual tasks that are not supported by information systems). The technique is therefore considered to be semi-automatic. Despite this, the technique is better than that of business experts redesigning business processes from scratch, since that solution is more time-consuming (Mutschler and Reichert, in press) and error-prone than semi-automatic techniques (Weerakkody and Currie, 2003). Furthermore, the manual business process redesign by business experts may ignore some business knowledge embedded in LISs.

The proposed technique is framed in MARBLE, a generic ADM-based framework to support business process mining, which was presented in a previous work (Pérez-Castillo et al., 2009a). ADM (Architecture-Driven Modernization) as defined by the OMG (Object Management Group) (OMG, 2009a), is a standard that advocates carrying out reengineering processes by considering model-driven development principles. The objective of MARBLE is to provide the highest abstraction level during the reverse engineering stage of ADM, i.e., the business knowledge that LISs depict. The MARBLE framework thus provides a solution to meet the demands detected by Khusidman and Ulrich (2007). It is also aligned with the SOA (Service-Oriented Architecture) research agenda developed by the SEI (Software Engineering Institute) (Lewis et al., 2010), which reports that business process recovery is needed to modernize LISs towards SOA systems.

All of the techniques framed in MARBLE are characterized by two key factors: (i) the set of knowledge sources, i.e., the software artifacts (e.g. source code, databases, user interfaces, etc.) used for recovering business processes; and (ii) the kind of reverse engineering technique (e.g. static or dynamic analysis of source code, program slicing, etc.) used to extract meaningful business knowledge. The proposed technique (i) considers legacy object-oriented code as the source of knowledge; and (ii) uses static (i.e. syntactical) analysis of source code as a reverse engineering technique. It is therefore possible to state that the technique supports static business process mining.

The proposed technique has been empirically validated using the supporting tool over the last two years by means of five industrial case studies involving five real-life LISs. In order to improve their rigor and validity, the case studies were conducted by following the formal protocol for case studies proposed by Runeson and Höst (2009). Each case study provided meaningful information concerning the effectiveness and efficiency of the proposed technique. This paper groups all that information and provides the meta-analysis of the results obtained for all case studies in order to attain strengthened conclusions, to evaluate relationships between the results obtained in each study, and to assess their effectiveness and efficiency measures. The empirical validation demonstrates that the technique enables the recovery of business processes with adequate accuracy levels, and that it is possible to do this in a scalable manner for large LISs.

The remainder of this paper is organized as follows. Section 2 summarizes the related work. Section 3 presents the proposed technique in detail. Section 4 presents the planning and execution of the family of case studies. Finally, Section 5 discusses the main conclusions attained from the meta-analysis, along with our future work.

2. Related work

Business knowledge preservation is not a new problem, and reverse engineering applied to the recovery of business processes

from LISs has been frequently addressed by both business experts and software academics. Some works address business knowledge recovery from LISs by statically analyzing various software artifacts. For instance, Ghose et al. (2007) use text-based queries executed in documentation to extract business knowledge. This proposal was only validated through an example. System databases are other artifacts that are used as input in business knowledge recovery based on static analysis. Paradauskas and Laurikaitis (2006) recover business knowledge by means of the inspection of the data stored in databases together with legacy application code. The authors of this work validated their proposal through the development of a controlled example. Wang et al. (2004) present a framework for business rule extraction from large LISs based on static program slicing. Program slicing is a reverse engineering technique consisting of decomposing the program into slices according to certain criteria (e.g. fragments of source code that uses a specific program variable). However, the framework is not able to represent the business knowledge recovered in an understandable and standard manner. This work was validated by means of a large complex financial legacy system, but this validation did not follow a formal protocol and no meta-analysis techniques were used to analyze the results in combination. do Nascimento et al. (2009) present a method for rewriting LISs based on business process management, but it provides a framework that consists of a manual business process recovery. None of these works follow model-driven development principles. Those works that present ad hoc solutions therefore have some formalization and automation problems. The validation of this proposal is based on model conformance checking through the π-calculus tool (Milner et al., 1992).

Zou et al. (2004) developed a model-driven framework based on a set of heuristic rules that syntactically analyze the source code and extract business processes. Although this work follows some model-driven development principles, it does not consider the ADM standard. This proposal was validated by means of a case study concerning an e-commerce system, but no formal protocol to conduct the case study was used. Another work which statically analyzes a database as the input artifact is provided by Pérez-Castillo et al. (2009b). This work proposes a model-driven reengineering framework to extract business logic from relational database schemas. This proposal was validated by means of a case study in which the framework was applied to a real legacy relational database of a research institute.

The only reverse engineering technique used in all of the aforementioned works is that of static analysis, which has the inconvenience that a lot of business knowledge is lost since it ignores all runtime knowledge. Dynamic analysis solves this problem since it facilitates consideration of the information derived by system execution (Cornelissen et al., 2009). For instance, Eisenbarth et al. (2003) present a business feature location technique based on dynamic analysis. This proposal neither follows model-driven principles nor extracts business knowledge according to any business process notation, and demonstrates its applicability by means of an example. Cai et al. (2009) propose an approach that combines the requirement reacquisition with a dynamic and static analysis technique to extract complex business processes that are triggered by external actors. This work was also validated by means of a non-formal case study. In addition, Di Francescomarino et al. (2009) recover business processes by dynamically analyzing the Web application GUI-forms, which are executed during a user's navigation. This work additionally provides a clustering algorithm for minimizing business processes obtained. However, this proposal is not aligned with model-driven development principles either. The validation in this case was made through a case study with an e-commerce system which was conducted in a non-formal way.

Other works addressing the dynamic approach provide process mining techniques that register event logs focusing on Process-Aware Information Systems (PAIS), i.e., process management systems such as Enterprise Resource Planning (ERP) or Customer Relationship Management (CRM). The nature of these systems (and particularly their process-awareness) facilitates the registration of events throughout process execution. Event logs depict the sequence of business process activities executed, and can be used to dynamically discover the current business processes. A considerable amount of work follows this approach. For example, van der Aalst et al. (2007) provides different business process mining algorithms to discover business process form event logs which were validated throughout a large industrial case study by applying different algorithms. Gunther et al. (2007) provides a generic import framework with which for obtaining event logs from different kinds of process-aware information systems. Ingvaldsen et al. (2008) focus on ERP systems to obtain event logs from the SAP's transaction data logs. However, neither work follows model-driven development principles. These two proposals were validated by applying them to real systems but both studies did not follow a formal protocol.

Dynamic analysis is more powerful and accurate than static analysis. The main drawback of static analysis is that it only focuses on static information retrieved by syntactically analyzing the existing source code or any other artifact. Moreover, there is much valuable knowledge that is known only at run time, for which dynamic analysis are more suitable. However, the applicability of dynamic analysis has some limitations since it usually requires source code modifications in order to aggregate traces to collect system execution information in an appropriate way (Canfora et al., 2011). In practice, source code modifications are not always possible since legacy information systems can be in the production stage. Another point of controversy related to the usage of dynamic analysis techniques in realistic environments is that it might be thought that a company or organization would not accept the use of an automatically modified version of its information system. For this reason, dynamic analysis techniques should ensure that the source code instrumentation is carried out without affecting the behavior of the original information system, i.e. in a non-invasive way. The main advantage of static analysis is its efficiency with regard to dynamic one.

Our proposal is framed in MARBLE, and is thus based on the ADM standard. It provides a semi-automatic reverse engineering technique based on static analysis of source code. The proposal is validated by a set of case studies as well as a meta-analysis study applied with the obtained results. Table 1 shows the comparison between the proposed technique and the other related work.

3. Static business process mining

The proposed business process recovery technique is framed in MARBLE. On the one hand, MARBLE depicts a general-purpose and extensible ADM-based framework for recovering business processes (cf. Section 3.1). MARBLE establishes the guidelines for various techniques whose objective is the same: retrieve business process models from legacy software artifacts. All particular MARBLE-based techniques can be considered as an instance of MARBLE and the commonalities of such techniques are: (i) the information is collected into and is used from standard KDM (Knowledge Discovery Metamodel) repositories and (ii) the information of KDM repositories is used to retrieve business process models.

On the other hand, the certain MARBLE-based technique presented in this paper establishes the three specific transformations needed to recover the business process (cf. Section 3.2).

Table 1
Comparison of work related to business process recovery.

	Our proposed technique*	Ghose et al. (2007)	Paradauskas and Laurikaitis (2006)	Wang et al. (2004)	do Nascimento et al. (2009)	Zou et al. (2004)	Pérez-Castillo et al. (2009b)	Eisenbarth et al. (2003)	Cai et al. (2009)	Di Francescomarino et al. (2009)	van der Aalst et al. (2007)	Gunther et al. (2007)	Ingvaldsen et al. (2008)
Static analysis	×	×	×	×		×	×						
Dynamic analysis		×						×	×	×	×	×	×
Recovery from event logs											×	×	×
Model-driven development principles	×				×	×	×						
ADM	×												

MARBLE-based techniques consist of three parts: (i) legacy software artifacts used to retrieve business process insights (e.g., source code, databases, etc.); (ii) technique to retrieve information from legacy software artifacts (e.g., static analysis vs. dynamic analysis); and (iii) the transformation or technique from KDM repository to business process models. The paper proposes a technique based on static analysis of object-oriented source code and manual post-intervention by business experts as the further valuable sources of knowledge.

3.1. MARBLE

Reengineering has normally been used to obtain new improved versions of aged LISs and, according to Chikofsky and Cross (1990), consists of three stages: reverse engineering, restructuring, and forward engineering. However, reengineering projects rarely reach the business abstraction level – they typically concentrate on the level of system design (Khusidman and Ulrich, 2007). According to Sneed (2005), reengineering usually lacks for formalism and standardization. Indeed, the majority of reengineering projects are usually carried out in an ad hoc manner. ADM helps to solve the formalism and standardization problems of reengineering. ADM-based processes deal with all artifacts involved in reengineering as models that conform to specific metamodels, and these processes also establish model transformations between models to deal with different abstraction levels throughout the three reengineering stages (Newcomb, 2005).

MARBLE (*Modernization Approach for Recovering Business processes from Legacy Systems*) (Pérez-Castillo et al., 2009a) is an ADM-based generic framework that facilitates business process recovery. All particular techniques framed in MARBLE specify the path of model transformations (cf. Section 3.2) needed to obtain each model in a specific level from a previous one. MARBLE defines four abstraction levels related to four different kinds of models: L0 to L3.

- *L0. Legacy information system*. This level represents the entire LIS that exists in the real world as a set of interrelated software artifacts: source code, user interfaces, databases, documentation, etc.
- *L1. Legacy information system models*. This level contains a set of models that can represent one or more software artifacts of the LIS at L0. These models are considered to be platform-specific models (PSM) since they represent different views or concerns of the systems from a technological point of view at a lower abstraction level.
- *L2. KDM model*. This level contains a single model that integrates all models of the previous L1 level. This level represents the entire LIS from a platform-independent model (PIM) at an intermediate abstraction level. MARBLE uses the KDM (Knowledge Discovery Metamodel) standard to represent this model. KDM is an ADM specification proposed by the OMG and has been recognized as the ISO 19506 standard (ISO/IEC, 2009). KDM provides a common repository structure that makes it possible to exchange information about all legacy software artifacts, and can be compared with the Unified Modeling Language (UML) standard; while UML is used to generate new code in a top-down manner, an ADM process involving KDM starts from the existing code and builds a higher level model in a bottom-up manner (Pérez-Castillo et al., in press).
- *L3. Business process models*. This level corresponds to models that represent the business processes recovered from the LIS. The models in this level are considered to be computer-independent models (CIM) since they depict a business view of the system at the highest abstraction level. In order to represent this kind of models, MARBLE uses the BPMN (Business Process Modeling and Notation) standard metamodel (OMG, 2009b), since it offers a

well-known graphical notation that is easily understood by both system analysts and business experts.

3.2. Proposed technique to recover business processes

The proposed business process recovery technique is framed in MARBLE, which is also automated by a supporting tool (Alarcos Research Group MARBLE Tool, 2011). The technique defines three model transformations and progressively presents a working example for explaining these transformations.

3.2.1. L0-to-L1 transformation

The first transformation takes into account legacy object-oriented code as the sole software artifact, since it is the artifact that embeds most business knowledge. This transformation thus obtains a code model that represents the source code, considering technological details. Indeed, this transformation must be tuned for each programming language, e.g. it is developed for Java-based systems. This transformation therefore consists of the static analysis of Java code files, which is carried out by means of a syntactical parser. As a result of the analysis, it builds code models according to the Java metamodel, which represents abstract syntax trees of code files.

An example concerning an information system supporting the product shipping of a reseller organization is now introduced to illustrate how the transformation works. This example focuses on a certain piece of source code: the '*ResellerControler*' class of the '*domain*' package (see Fig. 1, left). This class contains four methods that support four key functionalities of the system: (i) '*receiveOrder*' manages customers' order requests; (ii) '*checkInventory*' checks whether the products needed to fulfill a specific order are in stock; (iii) '*sendProducts*' manages the dispatch of the products ordered; and finally (iv) '*sendInvoice*' generates and sends the invoice to the customer.

According to the example, the L0-to-L1 transformation takes the Java source file (see Fig. 1, left) and obtains the abstract syntax tree that represents the java code model at L1 (see Fig. 1, right). The parser obtains a *CompilationUnit* element for each of the java source files analyzed, and then adds the *PackageDeclaration* and *ImportDeclaration* elements. The parser then generates a *ClassOrInterfaceDeclaration* element for the class, which contains a *ClassOrInterfaceBodyDeclaration* element for each method. Each method is represented through a *MethodDeclaration* with *ResultType*, *MethodDeclarator* and *Block* of *Statement* elements. The *Statement* element is, in turn, specialized into several kinds of elements: *ReturnStatement*, *IfStatement*, *Expression*, and so on. The right-hand side of Fig. 1 shows the elements used to represent the business logic implemented in the '*receiveOrder*' method.

3.2.2. L1-to-L2 transformation

The L1-to-L2 transformation is in charge of the transformation of the PSM models at L1 into a single PIM model at L2 according to the KDM metamodel. The KDM metamodel consists of several metamodel packages which are interrelated and organized into four abstraction layers. Each package defines a set of metaclasses with which to model a different view or concern of LISs. The KDM model at L2 considers the *code* and *action* KDM packages of which the *program element* layer of the KDM metamodel is composed. The *code* and *action* KDM packages provide a language-independent intermediate representation for various constructs determined by common programming languages. This transformation uses only these packages because legacy source code is the only artifact considered at L0. However, owing to the KDM representation, this information might, in the future, be integrated in a standardized manner with new valuable knowledge related to other software artifacts.
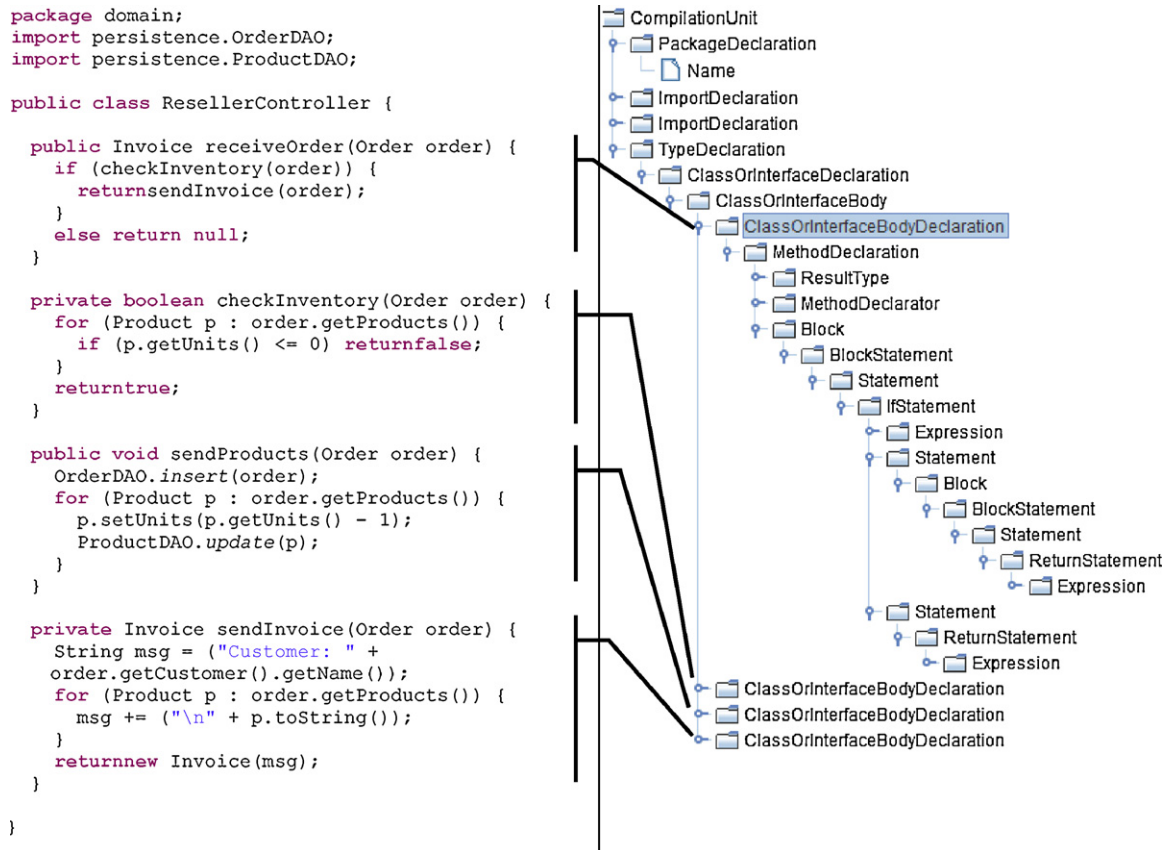
Fig. 1. An example of the L0-to-L1 transformation for a product shipping system.

In order to establish the model transformation between the code model at L1 and the KDM code model at L2, the technique uses the QVT (Query/Views/Transformation) language to implement the transformation. Owing to the fact that the structures of the Java metamodel (L1) and KDM code-action metamodel (L2) are very similar, the implementation of this transformation is almost immediate when using QVT (a declarative language), since it usually only needs to rename the metaclasses of both metamodels.

To continue with the example, Fig. 2 shows the KDM model obtained after the execution of the proposed QVT transformation for the product shipping system. The KDM model contains a *Package* instance named 'domain' with a nested *CompilationUnit* instance named 'Reseller'. The *CompilationUnit* is obtained from the *Class* instance of the Java code model at L1. The 'Reseller'*CompilationUnit* instance contains an instance of the *CallableUnit* metaclass for each Java method at L1. Each *CallableUnit* instance is also defined by means of different *CodeElement* and *ActionElement* instances. For example, the 'receiveOrder' method is modeled as a *Callable-Unit* containing (see Fig. 2): (i) an *EntryFlow* instance that defines the first KDM action in the unit (the first Java statement, since a model's sequentiality of actions is not clearly defined); (ii) a *Sig-nature* instance that defines the parameters of the unit; and finally (iii) an *ActionElement* instance that represents the *if* statement in the Java method. The remaining *CallableUnit* instances have a similar structure.

3.2.3. L2-to-L3 transformation

The L2-to-L3 transformation is the last transformation and obtains business process models at L3 from the previous KDM model at L2. This transformation uses the metamodel of the BPMN standard (OMG, 2009b) to represent the business process models. Fig. 3 shows the BPMN metamodel, which enables the

representation of business process diagrams as instances of the *BusinessProcessDiagram* metaclass. Each diagram can involve four kinds of elements: (i) flow object elements such as the *Activity*, *Task*, *Event* and *Gateway* metaclasses; (ii) connecting object elements such as *SequenceFlow*, *MessageFlow* and *Association*; (iii) artifact elements such as the *DataObject*, *group* and *Annotation* metaclasses; and (iv) swim lane elements which are used to group elements such as *Pool* and *Lane*.

This transformation consists of two steps: (i) a model transformation that obtains a set of preliminary business process models; and (ii) manual intervention by business experts that modifies the business processes obtained in order to improve them. Firstly, the model transformation establishes a set of business patterns (Pérez-Castillo et al., 2010a), which define which pieces of the source code (represented in a KDM code model) will be transformed into specific structures of a business process. Table 2 summarizes the set of proposed business patterns and briefly explains each pattern in natural language. The transformation is implemented through a set of QVT relations (Pérez-Castillo et al., 2010b), which support the pattern matching process. Each QVT relation searches for instances of the source structures defined by each pattern, and the QVT relations then enforce the creation (in the business process model) of instances of the target structures of the pattern for each input instance found in the KDM model.

Owing to space limitations, Fig. 4 shows only one QVT relation as an example (the *CallableUnit2Task* relation), although the complete transformation is available online (Pérez-Castillo, 2011). The *CallableUnit2Task* implements the *P2.Sequence* pattern, and transforms all *CallableUnit* instances (e.g. java methods) into *Task* instances, and invocations between them into sequence flows. This relation follows the well-known "*a callable unit-atask*" approach proposed by Zou et al. (2004). There are other choices that
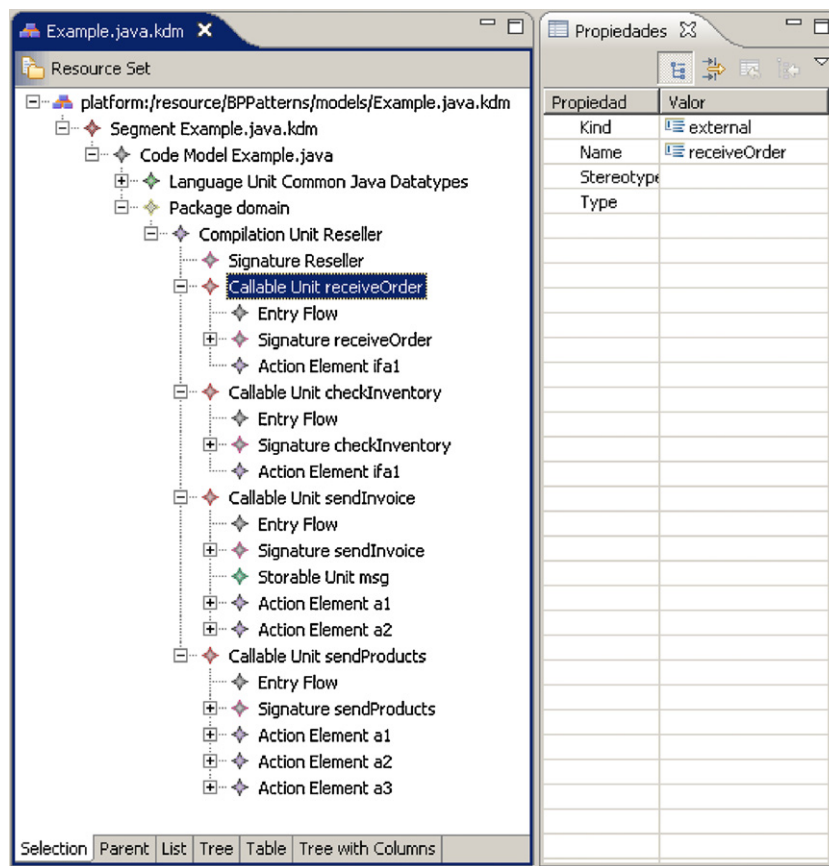
Fig. 2. The KDM model obtained after the proposed QVT transformation for the product shipping system.

transform methods and their respective calls into subprocess-task relationships (i.e. composite tasks). However, the "*a callable unit-a-task*" approach is better to solve problems regarding the method granularity, since it can treat all methods in the same manner and then, it can discard fine-grained methods using heuristics like removing the 'getters/setters' methods.

Finally, from the where clause, the *CallableUnit2Task* relation calls other relations that depend on the tasks created. For instance,
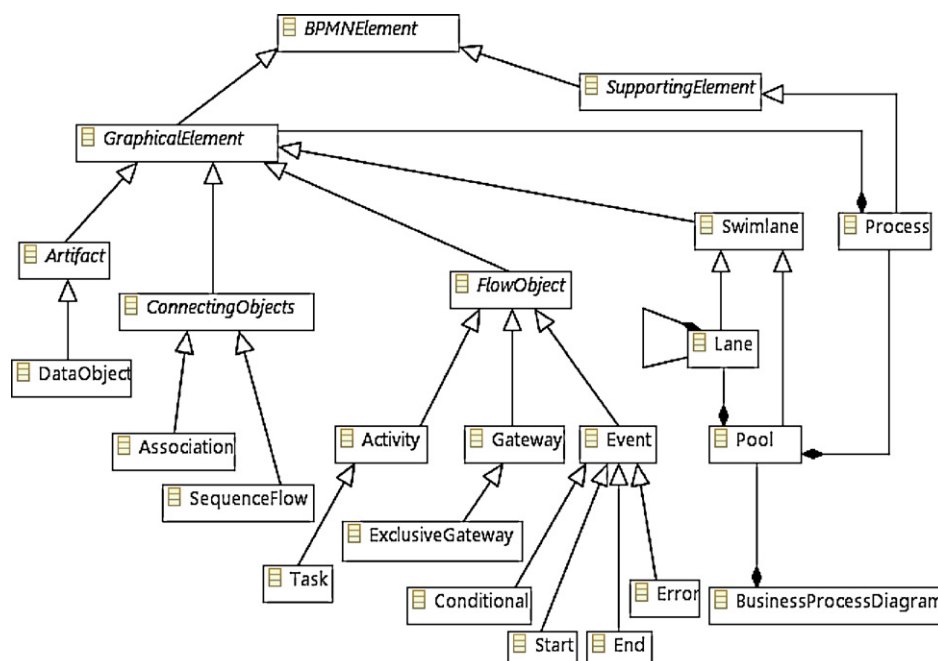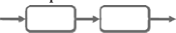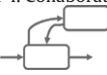
Fig. 3. Overview of the BPMN metamodel.

Table 2
Summary of the set of business patterns.

Pattern	Description
P1. BPD Skeleton	This pattern creates the root structure of the BP model. It creates a BP diagram for each KDM code model. Also, it builds a pool element with a nested process in the BP diagram for each package of the KDM code model
P2. Sequence	This pattern takes any callable piece of code from the KDM code model and maps them into tasks in the BP diagram. The sequence of calls to callable units is transformed into a set of sequence flows in the same order between the tasks built from the callable unit respectively
P3. Branching	This pattern transforms each conditional jump of the source code that has two mutually exclusive choices into an exclusive gateway and two different sequence flows in the BP model. Typically those exclusive conditional branches are related to the if . . . then . . . else or switch clauses in several programming languages. The exclusive gateway represents the condition that is evaluated and the two sequence flows represent two conditional transitions that depend on the value (true or false) of the evaluation
P4. Collaboration	Each call to external callable unit (i.e. API libraries or external components outside the legacy system) is transformed into an auxiliary task as well as two sequence flows: the first from the source task to the auxiliary task and the second returning to the source task
P5. Data input	This pattern builds a data object in the BP model for each input data within a callable unit in the KDM code model. Also, this pattern builds an association between the data objects and the task previously built from the callable unit. This pattern only considers as input data the parameters or arguments of the callable unit, but it does not consider the auxiliary variables within the callable unit
P6. Data output	Each piece of output data involved in a callable unit is transformed by means of this pattern into a data object as well as an association from the task (built from the callable unit) to the data object. This pattern excludes as output data the auxiliary and intermediate data in the body of the callable unit. The output data is the data returned by the callable unit or external data related to databases or files
P7. Start	The task building from the callable unit that starts the execution of any program or application of the legacy system is considered the initial task. A start event is built into the BP diagram and a sequence flow from this event to the initial task is also created
P8. Implicit termination	This pattern builds an end event in the BP model. Then, it creates sequence flows from 'end task' and those flows merge in the end event. A task is considered an 'end task' if this task does not have any outgoing sequence flow
P9. Conditional sequence	This pattern transforms each conditional call into a sequence flow fired under a conditional intermediate event through to the task related to the callable unit. This pattern makes it possible to create arbitrary cycles in the BP diagram
P10. Exception	Each call to callable unit under any exception is transformed into a task for the piece of source code that handles the exception as well as a sequence flow fired under an error intermediate event. Indeed, this pattern can be understood as a specialization of the previous pattern P9

the *WritesStorableUnit2DataObject* relation that implements the *P6.Data Output* pattern is called for each piece of data written by the callable unit.

Secondly, the L2-to-L3 transformation can be additionally aided by the manual post-intervention of business experts to refine the business processes obtained. This procedure is necessary in at least two cases: (i) when the technique has recovered some tasks related
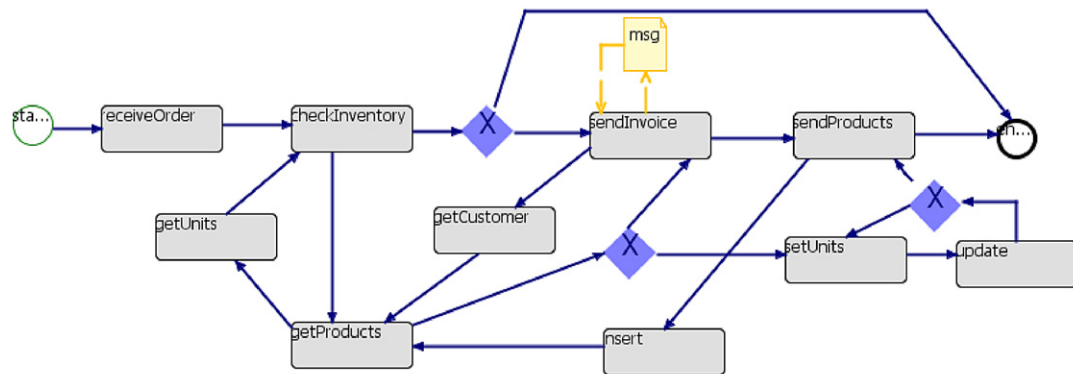
```
relation CallableUnit2Task {
    xName : String;
    checkonlydomain kdm m : code::CallableUnit {
        name = xName
    };
    enforcedomain bpmn pr : bpmn::Process {
        GraphicalElements = t :bpmn::Task {
            Name = xName,
            Status = bpmn::StatusType::None,
        }
    };
    where {
        ...
        m.codeElement->forAll (a: AbstractCodeElement | a.oclAsType
        (ActionElement).actionRelation->forAll (w:AbstractActionRelationship |
        (w.oclIsKindOf(Writes)) and w.oclAsType(Writes).to.oclIsKindOf
        (StorableUnit) implies WritesStorableUnit2DataObject (w, t, pr)));
        ...
    }
}
```

Fig. 4. A piece of QVT implementation of the L2-to-L3 transformation.

to the technical nature of legacy source code and these must be removed; and (ii) when there are manual business tasks that are not supported by information systems, and which must be added. It consequently proposes a semi-automatic technique. This could be considered as a critical point of our proposal, since it requires additional manual intervention by experts. However, business process redesign from scratch is a more time-consuming and difficult option than our proposal, which provides an initial meaningful understanding of current business processes. From an effort/cost viewpoint, any supporting tool during redesign increases the ability to redesign business processes, which is a factor to reduce the redesign effort (Mutschler and Reichert, in press).

To continue with the example, at the beginning the L2-to-L3 transformation obtains a first sketch of a business process diagram from the KDM model by means of the QVT transformation. Fig. 5(A) shows the graphical representation of the business process model obtained during the QVT transformation. This model contains 10 tasks in total, although only 4 tasks are related to the four *Callable-Unit* instances in the KDM model (see Fig. 2), which are obtained by applying the '*P2.Sequence*' pattern. The gateways are also created by applying the '*P3. Branching*' pattern. Moreover, 6 other tasks are also obtained by applying the '*P4.Collaboration*' pattern (see Table 2). This pattern is applied to the three *ActionElement* instances of the *CallableUnit* instance named '*sendProducts*' in the KDM model (see Fig. 2). These action elements represent calls to methods, which are not defined in the same Java source file (e.g. '*update*', '*setUnits*'

(A) Preliminary business process model obtained by means of the QVT transformation

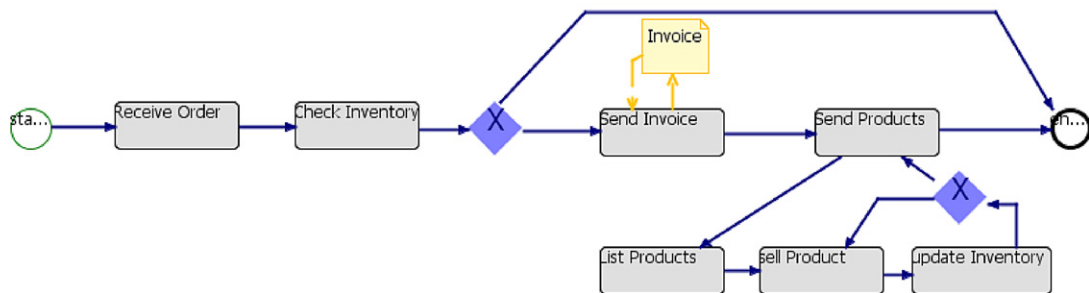(B) Business process model after manual intervention by business experts

Fig. 5. An example of business process models obtained for a product shipping system.

and 'insert'), and these calls are thus represented as external calls, which are transformed into six tasks with a round-trip sequence flow from the previous tasks.

Fig. 5(B) shows the business process model refined by business experts. There are three principal improvements:

- The three tasks obtained from external calls are removed, since the business experts consider that these tasks do not represent any of the organization's business activities, i.e. they are related to the technical dimension of the system.

```
Remove                  ['getCustomer];
Remove                  ['getUnits'];
Remove                  ['insert'];
```

- A gateway is added to merge the two branches opened after the first gateway.

```
Add[GATEWAY['sendProducts', 'gateway(CheckInventory)', END]];
```

- The remaining tasks and data objects are renamed by the business experts in order to fit the names to the organization's business activities.

```
Rename                  ['receiveOrder', 'Receive Order'];
Rename                  ['checkInventory', 'Check Inventory'];
Rename                  ['sendProducts', 'Send Products'];
Rename                  ['sendInvoice', 'Send Invoice'];
Rename                  ['getProducts', 'List Products'];
Rename                  ['update', 'Update Inventory'];
Rename                  ['setUnits', 'Sell Product'];
Rename                  ['msg', 'Invoice'];
```

Finally, Fig. 5(B) shows the business process model that represents a part of the organization's behavior through the business process recovery from the piece of source code presented at the beginning of this example (see Fig. 1, left).

4. Family of case studies

This section presents the family of case studies carried out over the last 2 years. The family was performed in five different information systems to validate the effectiveness and efficiency of the proposal. The case studies were carried out by following the formal protocol for case studies proposed by Runeson and Höst (2009) to improve their rigor and validity. The following sections present the stages of the protocol in detail: design, case selection, execution, data collection, analysis and interpretation, and an evaluation of its validity.

4.1. Design

The *object of study* is the business process mining technique, and the *purpose of this study* is to quantitatively evaluate the specific properties of the technique as regards their effectiveness and efficiency. A qualitative research about the usage of retrieved business process models to modernize legacy information systems could be very interesting. However, the evaluation of this usage is outside of the scope of the empirical evaluation since (i) such usage of business processes may be quite different in each case and (ii) the assessment of this usage would be probably carried out through surveys involving people that possibly do not have the same expertise level about business process management, which may imply biased results.

According to the object and purpose of the study, the main research question (*MQ*) is the following: *Can the technique properly obtain business processes from legacy systems?* In this question, the adverb '*properly*' is related to the feasibility and suitability of the technique, which means, in turn, business processes must be obtained in an effective and efficient manner. In order to discover

Table 3
Case study research questions.

Id	Research question	Evaluated properties
MQ	Can the technique properly obtain business processes from legacy information systems?	Feasibility/suitability
AQ1	Can the technique obtain business processes with adequate accuracy levels?	Effectiveness
AQ2	Is the proposed technique efficient?	Efficiency

whether *MQ* is true, Table 3 shows the additional research questions (AQ) that are identified from *MQ. AQ1* is established in order to evaluate whether the business processes recovered faithfully represent the organization's business behavior. That is, *AQ1* aims to evaluate the effectiveness of the procedure. Moreover, the additional research question *AQ2* (see Table 3) evaluates whether the proposed business process recovery procedure obtains business processes efficiently.

To answer the research questions, the study evaluates some properties by analyzing all retrieved business processes (see Table 3). For checking these properties, some measures are collected to provide quantitative answers to the proposed research questions.

Furthermore, the study follows an *embedded* design, since each study focuses on several analysis units within each single case. The analysis units are the different source code packages of each LIS, which is the independent variable of each study (i.e., the defined measures are evaluated for each analysis unit). Each source code package is represented in a code model at L1, in a KDM model at L2, and is eventually transformed into a business process model at L3.

The usage of code packages as the minimal unit to be transformed into a business process model could provide some processes with lower accuracy levels than other solutions considering correlation data set at the beginning of the process. However, the advantage of this heuristic solution is that it does not require the initial correlation information, and it automatically provides a preliminary business processes by statically analyzing the legacy source code. Those business processes can be used as the basis

to understand the entire business processes of the organization, which can be joined or split through manual post-intervention by business experts.

4.1.1. Evaluation of effectiveness

In order to evaluate the effectiveness of the technique according to *AQ1*, the study uses the *precision* and *recall* measures. These measures were designed for information retrieval scenarios, although they are usually applied to model recovery scenarios. Precision and recall measure the similarity between a mined business process *M* and a reference business process *R*. Precision indicates what proportion of *M* matches *R* (i.e., how exact *M* is), while recall indicates what proportion of *R* is present in *M* (i.e., how complete *M* is).

The study considers the *task* element as the score unit in order to apply these measures in a business process recovery scenario. It uses the opinions of business experts to discover which recovered tasks are or are not relevant. The precision measure (1) is therefore defined as the number of recovered relevant tasks divided by the total number of recovered tasks, while the recall measure (2) is defined as the number of recovered relevant tasks divided by the total number of relevant tasks. Although precision and recall are adequate, there is an inverse relationship between them. As a result, it is difficult to extract conclusions by using an isolated evaluation of these metrics. These measures are therefore usually combined into a single measure known as an *F*-measure (3), which consists of a weighted harmonic mean of both measures.
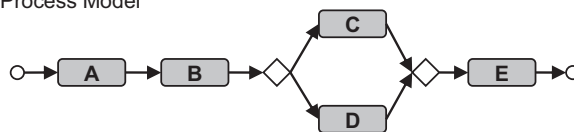
$$\text{Precision} = \frac{\{\text{recove rerelevant tasks}\}}{\{\text{recovered relevat tasks}\} + \{\text{recovered non-relevat tasks}\}} \quad (1)$$

$$\text{Recall} = \frac{\{\text{recove rerelevant tasks}\}}{\{\text{recovered relevat tasks}\} + \{\text{non-recovered relevat tasks}\}} \quad (2)$$
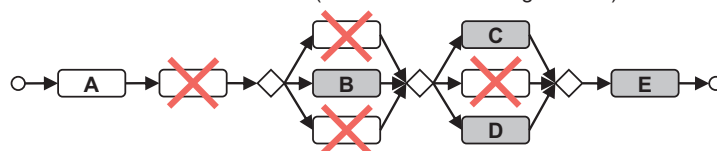
$$F\text{-measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

Fig. 6 and Table 4 shows some example business process models to enable the reader to understand how measures work. Fig. 6(A) shows a desirable source business process model with five tasks. Fig. 6(B) shows an example business process model retrieved with

(A) Source Business Process Model

(B) Retrieved Business Process Model (Low Precision and High Recall)

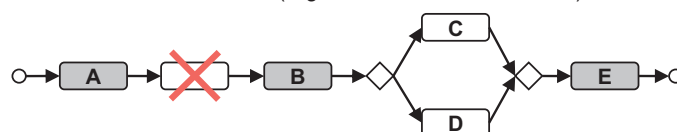(C) Retrieved Business Process Model (High Precision and Low Recall)

| A | Recovered Relevant Task | A | Non-Recovered Relevant Task | ✗ | Recovered Non-Relevant Task |

Fig. 6. Some business process models retrieved regarding a source model.

Table 4
Examples of measure values for some business process models.

Model	# Relevant tasks	# Recovered tasks	# Recovered relevant tasks	# Recovered non-relevant tasks	# Non-recovered relevant tasks	Precision	Recall	*F*-measure
A	5	5	5	0	0	1	1	1
B	5	8	4	4	1	0.5	0.8	0.61
C	5	4	3	1	2	0.75	0.6	0.67

four source tasks. One more task must thus be added by manual intervention. In addition, four other retrieved tasks are erroneous. This model presents low precision and high recall values since several non-relevant tasks were erroneously recovered (see Table 4). Finally, Fig. 6(C) provides another example business process model which was retrieved with three source tasks and one erroneous task. This model presents higher precision and lower recall than the previous model since the number of recovered non-relevant tasks is lower and the number of non-recovered relevant tasks is higher. The business process model retrieved in Fig. 6(C) is better than the model in Fig. 6(B) since the *F*-measure of (C) is higher (see Table 4) owing to the fact that the precision and recall values are more balanced (i.e., these values are closer).

Other important aspect to evaluate precision and recall is how a task is considered as a relevant task. Business experts check four conditions that should be met by the task under evaluation. The first condition specifies that the task must represent a real-life business operation within the organization. This condition is not evaluated by considering task names, since these names are inherited from legacy code and they may be biased regarding the real business activity names provided by business experts. The second condition ensures that all relevant tasks preceding the evaluated task must be recovered before the task under evaluation. In order to fulfill this condition, the predecessor tasks can be directly or indirectly connected to the task under evaluation, i.e., there could be non-relevant tasks intercalated between the evaluated task and their predecessor relevant tasks. In a similar manner, the third condition ensures that all subsequent tasks must be directly (or indirectly) recovered relevant tasks. Finally, the fourth condition specifies that all data objects related to the task under evaluation have been also recovered. The condition-based checking facilitates a final business process model after several iterations of evaluation.

4.1.2. Evaluation of efficiency

In order to evaluate the efficiency of the technique according to *AQ2*, the study employs the time spent on executing the whole transformation between MARBLE levels.

On the one hand, the *transformation time* is automatically measured in milliseconds by the tool developed. This measure is therefore analyzed with respect to the total number of elements built into each specific business process model (i.e., for each analysis unit). Both the number of elements and the transformation time are also considered as dependent variables in the study. A regression model is therefore established between the transformation time and the size of retrieved models to find out if the technique is or is not scalable to large legacy information systems.

On the other hand, the time derived by the manual intervention carried out by business experts is measured in order to know how

such intervention affects to the total transformation time. This is measured in seconds for each business process model retrieved and is also considered as a dependent variable of the study.

4.2. Case selection

The five LISs were selected by means of the evaluation of four case selection criteria (*C1* to *C4*). *C1* guarantees that the LIS selected is an information system that supports an organization's business operation. This criterion discards, for example, embedded systems or real-time systems. *C2* ensures that the selected system really is an LIS. This criterion considers the amount of modifications (from the time when the system was first released) which have altered the business processes supported by the system. *C3* ensures that the system is not a *toy program*, since it defines an arbitrary threshold of 20,000 lines of source code. Finally, *C4* guarantees that the system is based on the Java platform, since the tool supporting the technique was developed for Java-based systems.

After evaluating several available systems according to the aforementioned criteria, five systems were selected to be studied. Table 5 shows the name of each system, a brief description, the kind of architecture, and the size of the source code in thousands of lines.

4.3. Execution

The execution of the study is aided by the tool, which was developed to support the proposal. The case study procedure defines the followings steps:

1. After some meetings between the staff of the candidate organizations and researchers, the set of legacy information systems is selected according to the case selection criteria. At this point, business experts who will carry out the later manual verification in order to evaluate precision and recall measures. If possible, the case studies involve two experts with two different expertise levels. First, the chief operating officer (COO) who is responsible for ensuring that business operations are efficient and effective and therefore knows the actual business processes carried out by his company. Secondly, the chief information officer (CIO) who is responsible for the information technology and computer systems supporting business operation.
2. Each LIS under study is implanted in an experimental environment: the source code is deployed, the database schema is built through the database scripts, the initial data is loaded, etc.
3. The tool that supports the proposal is used to obtain the different business process models from the legacy source code. The

Table 5
Legacy information systems under study.

Id	Name	Description	Architecture	Size (KLOC)
S1	AELG-Members (AELG, 2009)	Supports the administration of an organization of authors	Desktop application	23.5
S2	Tap CRM (Source Tap, 2009)	Is a sales force automation tool for sales management	Web application	49.6
S3	VillasanteLab (Villasante, 2010)	Manages a laboratory of the water and waste industry	Web application	28.8
S4	XuntaEadmin (Enxenio Inc, 2009)	Supports the electronic administration of a Spanish regional ministry	E-government system	320.2
S5	SIXA (SIXA, 2009)	Is a learning management system	Web application	140.6

execution hardware environment consists of a computer with a RAM memory of 4 GB and two processors of 2.67 GHz each.

4. The first sketch of the business processes obtained through the model transformation is improved by business experts.

   4.1 They fit the preliminary business processes with the reality of the organization, i.e., they can add tasks that should have been recovered but were not recovered, or remove tasks that were erroneously recovered.

   4.2 After business expert intervention, the accuracy of the business process models is evaluated by comparing each preliminary business process and its related business process that has been enhanced by business experts. We obtain the value of the proposed measures such as precision and recall by scoring the differences between the preliminary and enhanced business processes.

   4.3 Steps 2, 3 and 4 are repeated for each LIS under study.

5. The key information related to the generation of business processes (cf. step 3), along with the business expert intervention (cf. step 4), is collected for each model according to the data collection plan (cf. Section 4.4).

6. The data collected in the previous step is analyzed and interpreted to draw conclusions in order to answer the research questions. Section 4.5 shows the outgoing results obtained in this case study, while Section 4.6 provides a meta-analysis of the analysis carried out in order to obtain strengthened conclusions.

4.4. Data collection

Table 6 summarizes the most relevant data collected during execution of the case studies. This table shows: (i) the business process identifier; (ii) the study identifier; (iii) the business process name; (iv) the number of tasks recovered (before manual intervention); (v) the number of recovered relevant tasks, i.e., tasks marked as correct by experts; (vi) the number of recovered non-relevant tasks, i.e., tasks removed since they did not represent a business activity; (vii) the number of non-recovered relevant tasks, i.e., tasks added by experts; (viii) precision and (ix) recall values for each final business process; (x) the harmonic mean between them; (xi) the total number of tasks for each final business process; (xii) the transformation time in milliseconds; and finally (xiii) the time spent on manual post-intervention in seconds. Table 6 also shows the mean values for each study.

4.5. Analysis and interpretation

The collected data were analyzed to obtain the evidence chains from data to answer the research questions. In order to answer the question *AQ1*, Fig. 7 shows the box chart for precision and recall measures. The means of the distributions of the precision measure (0.51, 0.53, 0.66, 0.64, and 0.54) are always lower than the recall means (0.77, 0.83, 0.91, 0.65, and 0.66). Higher recall values signify that the proposed technique recovers very complete processes (i.e., it recovers most of the tasks from the current business processes). However, the lower precision values signify that the technique is imprecise (i.e., the number of recovered non-relevant tasks is very high) (see Fig. 7). The significant amount of non-relevant tasks is owing to the fact that some tasks are basically obtained from the source code and they are related to the technical nature, and do not therefore represent any piece of the embedded business knowledge.

The transformation time was also analyzed in order to answer *AQ2*. The means of the transformation time for the five systems were 24, 2143, 8589, 8931 and 2315 ms. These values were different because the average of the process size for each study was quite different: 52, 33, 107, 72 and 33 tasks (see Table 6). These time values seem feasible for the selected cases, since the size of
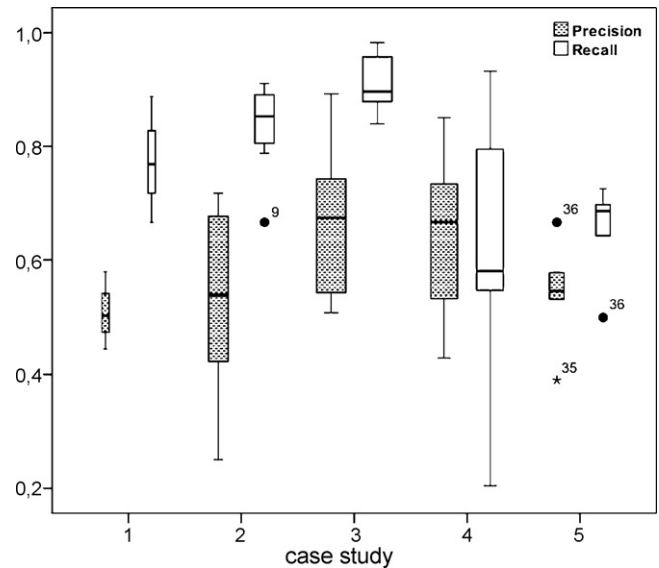

Fig. 7. Box chart of the precision and recall measures for each study.

the systems is above 20 *KLOC*. Nevertheless, the scalability of the procedure must be evaluated. Under the hypothesis that the time complexity of the procedure is theoretically linear (i.e., $O(n)$ considering 'n' as the number of tasks in a business process model) a linear regression model is established to check it and discover whether the proposal is therefore scalable.

The linear regression model considers the transformation time as a dependent variable and the size of the business processes as the independent variable. Fig. 8 shows the scatter charts of size/time for the five studies, along with their regression lines. Fig. 8 shows the regression line by grouping all values. The total regression line presents a positive linear relationship with $R^2 = 0.46$. The correlation coefficient R^2 (between -1 and 1) is the degree to which the real values of the dependent variable are close to the predicted values. The R^2 value obtained is a discrete result, but it is obtained as a consequence of lower R^2 values for individual studies like the first and fourth study, which have a few points to obtain interpolation regression lines with high statistical power level. In any case, the proposed linear regression model is suitable to explain the data obtained in this study, i.e., there is no quadratic or exponential relationship between the transformation time and the model size. The increase in time for larger systems will consequently be linear, and the time will thus be assumable. In conclusion, *AQ2* can therefore be answered as true, and the main research question *MQ* is also answered as true, i.e., the proposed business process recovery procedure makes it possible to obtain business process models from LISs in an effective and efficient manner.

Owing to the fact that the proposed technique is semi-automatic, the manual effort made by the business experts must be also analyzed, at least from a qualitative point of view. The selected business expert took an average of 362 s (6 min) for each business process model modified from the first sketch of the models retrieved by the tool. The manual time spent on this study is greater than the computational time. In fact, the manual intervention stage could be the bottleneck of the technique when it deals with large and complex system (see Fig. 9). Thereby, precision values around 55% signify that the retrieved models contains quite noise, which will imply a great effort during manual post-intervention. However, the manual time related to the proposed semi-automatic technique would be less than the time spent on redesigning a manual business process from scratch (Mutschler and Reichert, in press). The fault tolerance would be better with the proposed technique since

Table 6
Business process data collection for all case studies.

Subject ID	Study ID	Business process model name	# Rec tasks	# Rec rel tasks	# Rec non-rel tasks	# Non-rec rel tasks	Precision	Recall	*F*-measure	Size	Transf. time (ms)	Manual time (s)
1	S1	Category mgmt.	69	40	29	12	0.580	0.769	0.661	52	22	690
2	S1	Author mgmt.	141	71	70	9	0.504	0.888	0.643	80	35	520
3	S1	Reporting	36	16	20	8	0.444	0.667	0.533	24	14	240
		S1 mean	82	42	40	10	0.51	0.77	0.61	52	24	483
4	S2	Security mgmt.	69	30	39	4	0.435	0.882	0.583	34	1566	120
5	S2	Administration	45	26	19	7	0.578	0.788	0.667	33	630	300
6	S2	Chemical analysis mgmt.	80	51	29	5	0.638	0.911	0.750	56	3974	660
7	S2	Chemical calibration mgmt.	39	28	11	6	0.718	0.824	0.767	34	1184	540
8	S2	User mgmt.	64	16	48	3	0.250	0.842	0.386	19	782	180
9	S2	Chemical dilution mgmt.	39	16	23	8	0.410	0.667	0.508	24	1123	480
10	S2	Reporting	53	38	15	6	0.717	0.864	0.784	44	4981	120
11	S2	District mgmt.	36	18	18	2	0.500	0.900	0.643	20	2907	90
		S2 mean	53	28	25	5	0.53	0.83	0.64	33	2143	311
12	S3	Administration	134	69	65	10	0.515	0.873	0.648	79	4771	150
13	S3	Social-protection flat mgmt.	97	64	33	6	0.660	0.914	0.766	70	5895	280
14	S3	Document mgmt.	33	23	10	3	0.697	0.885	0.780	26	2724	540
15	S3	House rental mgmt.	290	221	69	9	0.762	0.961	0.850	230	23383	700
16	S3	Document renovation	204	169	35	3	0.828	0.983	0.899	172	10738	230
17	S3	House applicant mgmt.	213	190	23	9	0.892	0.955	0.922	199	13210	540
18	S3	Personal file mgmt.	134	74	60	9	0.552	0.892	0.682	83	5249	320
19	S3	Rural house mgmt.	118	63	55	12	0.534	0.840	0.653	75	10383	310
20	S3	Developer mgmt.	146	87	59	15	0.596	0.853	0.702	102	7034	280
21	S3	Renovation mgmt.	132	91	41	3	0.689	0.968	0.805	94	10086	430
22	S3	Emancipation grant mgmt.	122	62	60	7	0.508	0.899	0.649	69	5301	380
23	S3	Second-hand house mgmt.	105	76	29	9	0.724	0.894	0.800	85	4302	350
		S3 mean	144	99	45	8	0.66	0.91	0.76	107	8590	376
24	S4	Schedule mgmt.	104	73	31	16	0.702	0.820	0.756	89	7549	450
25	S4	Qualification mgmt.	73	56	17	5	0.767	0.918	0.836	61	14670	590
26	S4	Class group mgmt.	66	44	22	34	0.667	0.564	0.611	78	3218	550
27	S4	Evaluation	33	21	12	82	0.636	0.204	0.309	103	2862	430
28	S4	Remark mgmt.	87	74	13	22	0.851	0.771	0.809	96	17952	590
29	S4	Reporting	147	98	49	36	0.667	0.731	0.698	134	18289	230
30	S4	Notice mgmt.	45	25	20	18	0.556	0.581	0.568	43	1628	310
31	S4	Permission mgmt.	47	24	23	18	0.511	0.571	0.539	42	2411	290
32	S4	Student mgmt.	74	36	38	32	0.486	0.529	0.507	68	25261	460
33	S4	Subject mgmt.	21	9	12	8	0.429	0.529	0.474	17	862	520
34	S4	Teacher mgmt.	70	55	15	4	0.786	0.932	0.853	59	3547	400
		S4 mean	70	47	23	25	0.64	0.65	0.63	72	8932	438
35	S5	Provider Mgmt.	59	23	36	10	0.390	0.697	0.500	33	1905	130
36	S5	Product mgmt.	6	4	2	4	0.667	0.500	0.571	8	172	240
37	S5	Event mgmt.	47	25	22	12	0.532	0.676	0.595	37	1252	180
38	S5	Sending mgmt.	64	37	27	14	0.578	0.725	0.643	51	1674	240
39	S5	Reporting	33	18	15	10	0.545	0.643	0.590	28	7471	150
40	S5	Relationship mgmt.	55	30	25	13	0.545	0.698	0.612	43	1419	270
		S5 mean	44	23	21	11	0.54	0.66	0.59	33	2316	202
		Total mean	85.8	54.8	31.0	12.6	0.601	0.775	0.664	67.4	5810.9	362
		Standard deviation	58.2	47.3	18.0	13.9	0.138	0.166	0.138	48.3	6430.0	172

1: R^2 Linear = 0,981
2: R^2 Linear = 0,401
3: R^2 Linear = 0,773
4: R^2 Linear = 0,229
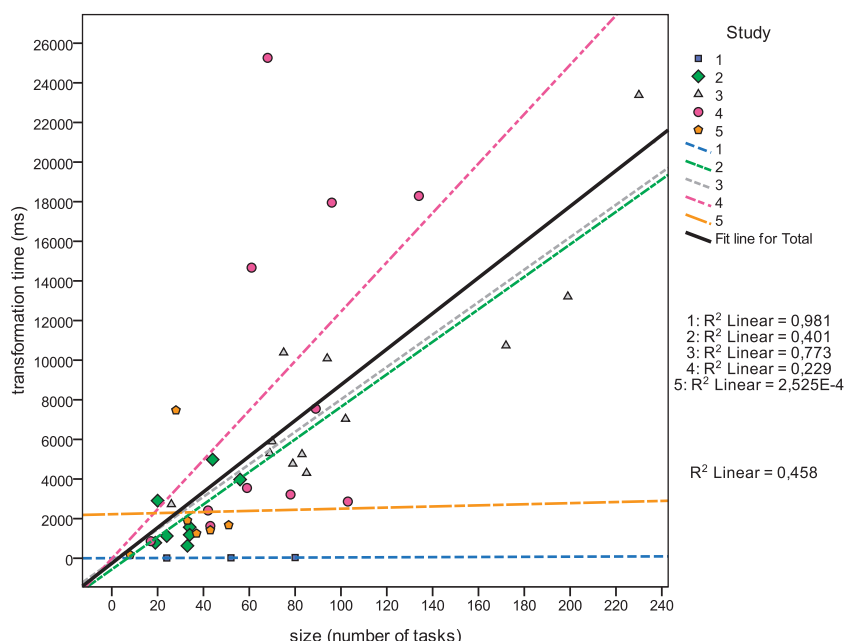5: R^2 Linear = 2,525E-4

R^2 Linear = 0,458

Fig. 8. The size/time scatter chart considering a linear regression model.

it provides a first sketch of the actual business processes. Furthermore, redesign from scratch by manual solutions has at least two drawbacks. Firstly, manual solutions do not consider the business knowledge that is embedded in LISs, and which is not present in any other artifact. Secondly, business process redesign copes with the numerous technological, organizational and project-driven cost factors related to the context of BPM projects. For example, manual redesign can be influenced by intangible impact factors like available process knowledge or end user fears (Mutschler and Reichert, in press).

The manual intervention time, which is high, could be reduced by adding methods to aid decision-making of business experts. For example, the tool could analyze the business processes and provides some suggestions to improve the obtained models. Particularly, such analysis could determine similar fragments in two different business processes, thus it may suggest joining those

Fig. 9. Relationship between manual and computational time in the five case studies.

processes. Also, the analysis could evaluate metrics such as the number of tasks and the intricacy degree (ratio of sequence flows regarding tasks) and suggests splitting a process into two or more processes according to the subsets of data that some process fragments manage. As a result, the tool will provide a prioritized list of candidate actions, for which business experts could select the most suitable actions and/or provides additional modifications.

4.6. Meta-analysis study

Several statistical methods (e.g., meta-analysis, significance level combination or vote counting) allow researchers to accumulate and interpret a set of results obtained through different empirical studies that are inter-related because they check similar hypotheses (Hedges and Olkin, 1985; Wolf, 1986). In the study presented here, we have used meta-analysis to strengthen our conclusions.

While simple analysis attempts to evaluate a sole study in isolation, meta-analysis combines results of several ones. Meta-analysis is a set of statistical techniques that combine the different effect size measures or treatment effect of various individual studies. There are several metrics to obtain this value, e.g. the means difference and the correlation coefficients, among others (Hedges and Olkin, 1985). The objective is to obtain a global treatment effect of all the cases studies.

As effect size measures may come from different environments and heterogeneous studies or experiments, Meta-analysis first obtains standardized measures of each one. After that, the global effect size is obtained as a weighted average of standardized measures, in which the most commonly used weights are the sample size or the standard deviation. Finally, together with the estimation of the global effect size, meta-analysis provides an estimated confidence interval and a *p*-value which is used to decide on the meta-analysis hypotheses. Most meta-analysis techniques are automated by tools. The meta-analysis presented in this work has been obtained by using the *Meta-Analysis v2* tool (Biostat Inc Comprehensive Meta-Analysis v2, 2006).

The objective of the meta-analysis presented here is the evaluation of the difference in means for the recall and precision measures obtained in each case study. Once the overall effect size is
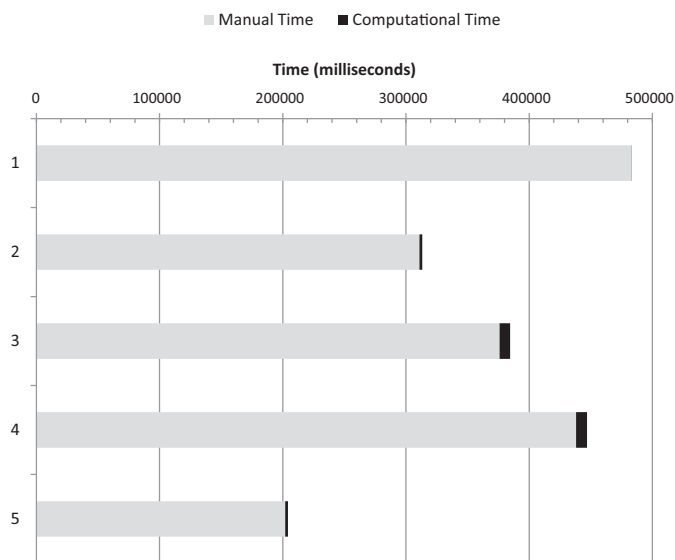
Recall vs Precision

Study name	Statistics for each study					Std diff in means and 95% CI
	Std diff in means	Standard error	Lower limit	Upper limit	p-Value	
S1. AELG-Members	-2,877	1,165	-5,160	-0,594	0,014	
S2. Tap CRM	-2,403	0,656	-3,689	-1,117	0,000	
S3. VillasanteLab	-2,503	0,545	-3,571	-1,434	0,000	
S4. XuntaEadmin	-0,050	0,426	-0,886	0,786	0,907	
S5. SIXA	-1,333	0,638	-2,584	-0,082	0,037	
	-1,368	0,264	-1,885	-0,851	0,000	
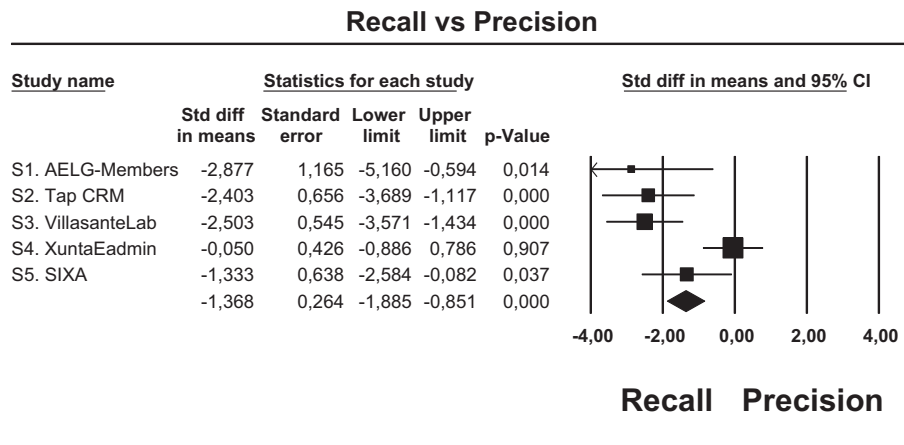
Recall   Precision

Fig. 10. Meta-analysis results and forest plot (recall vs. precision).

calculated, we can provide a confidence interval or a *p*-value which allows us to decide on the meta-analysis hypotheses, such as those which can be found in other empirical software engineering works (Cruz-Lemus et al., 2009; Dybå et al., 2007). Our meta-analysis hypothesis can be stated as: *recall is as accurate as precision in our technique for recovering business processes (H_0).*

Fig. 10 shows the forest plot after applying the meta-analysis. The left-side of the plot lists the names of the studies. The right-side of the plot shows the measure of effect, i.e., difference of means, for each of these studies (represented by a square) incorporating confidence intervals (represented by horizontal lines). The area of each square is proportional to the study's weight in the meta-analysis. Finally, the overall meta-analyzed measure of effect is represented as a diamond.

The results obtained (see Fig. 10) allow us to reject this hypothesis, as they indicate that recall works more efficiently than precision. These results agree with those stated in the previous sub-section and they are statistically significant at a level of 0.05, so we can conclude that the idea of recall being more accurate than precision in our business technique is statistically confirmed.

The result obtained for recall and precision measures is usual since there is an inverse relationship between both measures (see Fig. 11). The precision value should, ideally, always be 1 for any recall value, but this is not possible in practice. In fact, the proposed method could increase its Precision value by recovering fewer tasks since the precision measure has the total number of recovered task in the denominator. To maintain the higher Recall value, the reduction should only focus on non-relevant tasks to achieve a higher

Precision while Recall is as higher as possible. It is however a hard task and some relevant tasks will probably be also discarded. As a result, while Recall has been reduced a little bit, Precision has been increased much more regarding the reduction of Recall. This hypothetical result would be more desirable than the obtained result, since the Precision and Recall values would be more balanced.

In order to jointly evaluate both measures, the *F*-measure values, which were 0.61, 0.64, 0.76, 0.63 and 0.59 for the five studies (see Table 6), were also analyzed. These values were compared with a reference value of around 0.55, which is considered in other model recovery experiences (Lucrédio et al., 2008). The values obtained for all studies were clearly above 55%. Moreover, the total means obtained by grouping all studies were precision = 0.60, recall = 0.77 and *F*-measure = 0.66 (see Table 6), which were above the reference value. As a result, *AQ1* can be positively answered, i.e., the proposed technique recovers business processes from LISs with adequate accuracy levels. However, the *F*-measure value might be slightly improved by obtaining more balanced precision and recall values.

4.7. Evaluation of validity

This stage evaluates whether the results are true and not biased for the whole population to which we wish to generalize the results. This section thus shows the threats to the validity of the family of case studies.

According to the *internal validity*, an average population of 40 business process models was recovered, and it is thus possible to obtain statistically representative results. Nevertheless, the study may be replicated by using more systems to attain a larger population. In addition, there are two determining factors related to obtaining the results presented here: (i) the supporting tool used to obtain the business processes could be a factor that affects the transformation time values, and (ii) if the study is replicated with other cases involving different business experts, the measure values might have some deviations, since each expert provides his/her subjective viewpoint. The replication of the study using different configurations is a possible means to mitigate these threats.

Moreover, according to the *construct validity*, the selected measures were adequate to answer the research questions in an appropriate manner. Both precision and recall were reused from the information retrieval field, in which these metrics have an adequate maturity level. However, the way in which such measures are evaluated could be a threat, since these consider tasks as the sole scoring element. To mitigate this threat, alternative mechanisms could be considered such as the combination of various scoring elements at the same time (e.g., tasks and sequence flows).
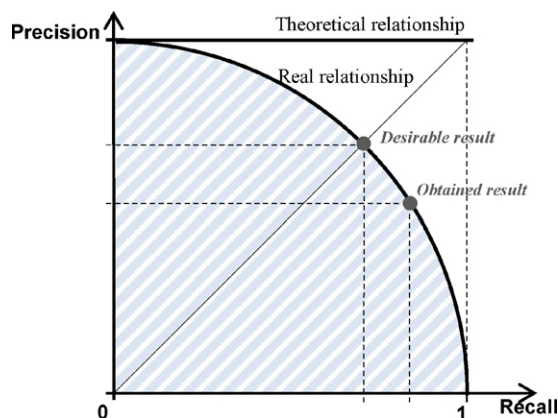

Fig. 11. Relationship between precision and recall measures and obtained/desirable results.

Anyway, tasks are, or are not, considered as relevant tasks by assessing their accuracy as well as their related sequence flows, data objects, and so on.

Finally, *external validity* is concerned with the generalization of the results. This study considers, as the whole population, traditional LISs following the object-oriented paradigm. The results obtained could therefore be strictly generalized to this population. However, the specific platform of the selected case is a threat that should be noted. This threat could be mitigated by replicating the study using systems that follow different platforms (not solely object-oriented).

5. Conclusions and future work

This paper has addressed the problem of business process recovery from legacy information systems in order to preserve valuable business knowledge. The preservation of this knowledge facilitates the alignment between the actual business behavior in an organization and their information systems. In addition, business knowledge preservation allows maintainers to modernize legacy information systems, extending their lifecycles and therefore improving the ROI. This paper provides a technique with which to recover business process, and which is framed in MARBLE, an ADM-based framework. The technique is characterized by three main features: (i) it focuses on legacy object-oriented code as the main source of knowledge, (ii) it uses static analysis as a reverse engineering technique to extract the information needed; and (iii) it follows the model-driven development principles, i.e., it considers different models at different abstraction levels and a set of model transformations between them. A supporting tool is additionally provided in order to automate the technique and facilitate its adoption.

In order to validate the proposal, this paper presents a family of industrial case studies, which were carried out over 2 years with five real-life legacy information systems. The case studies evaluate the effectiveness and efficiency of the technique. The effectiveness is evaluated using precision and recall measures. These metrics are appropriate to discover the accuracy (precision) and completeness (recall) of each of the business processes recovered. Moreover, in order to evaluate the efficiency, the study evaluates the recovery time with regard to the size of each business process model. It therefore checks the scalability of the technique to large legacy information systems.

The results obtained from the study were also meta-analyzed to obtain strengthened conclusions. The result obtained shows that the technique is suitable to recover business processes in an effective and efficient manner. However, according to the effectiveness, the recall values were better than the precision values. We believe that these results were obtained because much of the work was, in several cases, basically recovered from technical code.

Our future work will address the threats to validity identified. The study will be replicated with more legacy information systems based on other platforms or languages in order to compare and generalize (if possible) the results obtained. Besides improving the empirical validation, the technique will also be improved by incorporating the dynamic analysis in order to extract more and valuable business knowledge during system execution. The final objective is to obtain more balanced values of recall and precision, although the *F*-measure values must be simultaneously preserved.

References

Alarcos Research Group MARBLE Tool, 2011. Available from: http://www.business-processarcheology.org/downloads/MARBLE.rar [cited 10.06.11].

Biostat Inc Comprehensive Meta-Analysis v2, 2006, available from: http://www.meta-analysis.com/ (cited 17.08.10).

Cai, Z., Yang, X., Wang, W.,2009. Business process recovery for system maintenance—an empirical approach. In: 25th International Conference on Software Maintenance (ICSM'09). IEEE Computer Society, Edmonton, Alberta, Canada, pp. 399–402.

Canfora, G., Di Penta, M., Cerulo, L., 2011. Achievements and challenges in software reverse engineering. Communications in ACM 54 (4), 142–151.

Chang, J.F., 2006. Business Process Management Systems: Strategy and Implementation. Auerbach Publications.

Chikofsky, E.J., Cross, J.H., 1990. Reverse engineering and design recovery: a taxonomy. IEEE Software 7 (1), 13–17.

Cornelissen, B., et al., 2009. A systematic survey of program comprehension through dynamic analysis. IEEE Transactions on Software Engineering 35 (5), 684–702.

Cruz-Lemus, J.A., et al., 2009. Assessing the understandability of UML statechart diagrams with composite states—a family of empirical studies. Empirical Software Engineering 14, 685–719.

Di Francescomarino, C., Marchetto, A., Tonella, P.,2009. Reverse engineering of business processes exposed as web applications. In: 13th European Conference on Software Maintenance and Reengineering (CSMR'09). IEEE Computer Society, Fraunhofer IESE, Kaiserslautern, Germany, pp. 139–148.

do Nascimento, G.S., et al.,2009. A method for rewriting legacy systems using business process management technology. In: 11th International Conference on Enterprise Information Systems (ICEIS'09). INSTICC, Milan, Italy, pp. 57–62.

Dybå, T., et al., 2007. Are two heads better than one? on the effectiveness of pair programming. IEEE Software 24 (6), 10–13.

Eisenbarth, T., Koschke, R., Simon, D., 2003. Locating features in source code. IEEE Trans. Softw. Eng. 29 (3), 210–224.

Enxenio Inc Enxenio web site, 2009. http://www.enxenio.es (cited 01.03.09).

Enxenio Inc. AELG-Members, 2009. http://www.aelg.org/ (cited 18.08.2011).

Enxenio Inc SIXA, 2009. http://www.sixa.es/es (cited 18.08.11).

Ghose, A., Koliadis, G., Chueng, A., 2007. Process Discovery from Model and Text Artefacts. In: IEEE Congress on Services (Services'07), Salt Lake City, UT.

Günther, C.W., van der Aalst, W.M.P., 2007. A generic import framework for process event logs. In: Business Process Intelligence Workshop (BPI'06), LNCS 4103, pp. 81–92.

Hedges, L.V., Olkin, I., 1985. Statistical Methods for Meta-analysis. Academia Press.

Heuvel, W.-J.v.d., 2006. Aligning Modern Business Processes and Legacy Systems: A Component-Based Perspective (Cooperative Information Systems). The MIT Press.

Ingvaldsen, J.E., Gulla, J.A., 2008. Preprocessing Support for Large Scale Process Mining of SAP Transactions. In: Business Process Intelligence Workshop (BPI'07), LNCS 4928, pp. 30–41.

ISO/IEC, ISO/IEC DIS 19506, 2009. Knowledge Discovery Meta-model (KDM), v1.1 (Architecture-Driven Modernization). ISO/IEC, p. 302, http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?ics1=35&ics2=080&ics3=&csnumber=32625.

Khusidman, V., Ulrich, W., 2007. Architecture-Driven Modernization: Transforming the Enterprise. DRAFT V.5. OMG, p. 7, http://www.omg.org/docs/admtf/07-12-01.pdf.

Koskinen, J., et al., 2004. Estimation of the Business Value of Software Modernizations. Information Technology Research Institute, University of Jyväskylä.

Lehman, M.M., 1984. Program evolution. Information Processing & Management 20 (1–2), 19–36.

Lewis, G.A., Smith, D.B., Kontogiannis, K., 2010. A Research Agenda for Service-Oriented Architecture (SOA): Maintenance and Evolution of Service-Oriented Systems. Software Engineering Institute, p. 40.

Lucrédio, D., Fortes, R.P.M., Whittle, J., 2008. MOOGLE: A Model Search Engine, in 11th international conference on Model Driven Engineering Languages and Systems. Springer-Verlag, Toulouse, France, pp. 296–310.

Milner, R., Parrow, J., Walker, D., 1992. A calculus of mobile processes. I. Inf. Comput. 100 (1), 1–40.

Mutschler, B., Reichert, M., 2012. Understanding the costs of business process management technology. Advances in Business Process Management, http://dbis.eprints.uni-ulm.de/742/. Access Date: 21/12/2011.

Newcomb, P.,2005. Architecture-driven modernization (ADM). In: Proceedings of the 12th Working Conference on Reverse Engineering. IEEE Computer Society.

OMG, 2009a. Architecture-Driven Modernization Standards Roadmap. February 2009; Available from: http://adm.omg.org/ADMTF%20Roadmap.pdf (cited 29.10.09).

OMG, 2009b. Business Process Model and Notation (BPMN) 2.0. Object Management Group, p. 496.

Paradauskas, B., Laurikaitis, A., 2006. Business Knowledge Extraction from Legacy Informations Systems. Journal of Information Technology and Control 35 (3), 214–221.

Paradauskas, B., Laurikaitis, A., 2011. Extracting Conceptual Data Specifications from Legacy Information Systems. Elektronika ir elektrotechnika 107 (1), 46–50.

Pérez-Castillo, R. KDM to BPMN Transformation implemented in QVT Relations. 2011 26/04/2011, available from: http://alarcos.esi.uclm.es/per/rpdelcastillo/modeltransformations/KDM2BPMN.htm (cited 02.05.2011).

Pérez-Castillo, R., et al.,2009a. MARBLE: A Modernization Approach for Recovering Business Processes from Legacy Systems. In: International Workshop on Reverse Engineering Models from Software Artifacts (REM'09). Simula Research Laboratory Reports, Lille, France, pp. 17–20.

Pérez-Castillo, R., et al., 2009b. PRECISO: A Reengineering Process and a Tool for Database Modernisation through Web Services. In: 24th Annual ACM Symposium on Applied Computing (SAC'09), Hawaii, USA, pp. 2126–2133.

Pérez-Castillo, R., et al.,2010a. Business Process Patterns for Software Archeology. In: 25th Annual ACM Symposium on Applied Computing (SAC'10). ACM, Sierre, Switzerland, pp. 165–166.

Pérez-Castillo, R., García-Rodríguez de Guzmán, I., Piattini, M.,2010b. Implementing Business Process Recovery Patterns through QVT Transformations. In: International Conference on Model Transformation (ICMT'10). Springer-Verlag, Málaga, Spain, pp. 168–183.

Pérez-Castillo, R., García Rodríguez de Guzmán, I., Piattini, M., 2011. Knowledge discovery metamodel—ISO/IEC 19506: a standard to modernize legacy systems. Computer Standards & Interfaces Journal 33 (6), 519–532.

Runeson, P., Höst, M., 2009. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14 (2), 131–164.

Sneed, H.M.,2005. Estimating the Costs of a Reengineering Project. In: Proceedings of the 12th Working Conference on Reverse Engineering. IEEE Computer Society, pp. 111–119.

Source Tap, Source Tap CRM, 2009. http://sourcetapcrm.sourceforge.net/.

van der Aalst, W., Reijers, H., Weijters, A., 2007. Business process mining: an industrial application. Information Systems 32 (5), 713–732.

Villasante, J.M., 2010. Laboratorio de Análisis J.M. Villasante. http://www.laboratoriovillasante.com/index.php [cited 18.08.11].

Visaggio, G., 2001. Ageing of a data-intensive legacy system: symptoms and remedies. Journal of Software Maintenance 13 (5), 281–308.

Wang, X., et al.,2004. Business rules extraction from large legacy systems. In: Proceedings of the Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR'04). IEEE Computer Society.

Weerakkody, V., Currie, W.,2003. Integrating business process reengineering with information systems development: issues and implications. In: Proceedings of the 2003 international conference on Business process management. Springer-Verlag, Eindhoven, The Netherlands, pp. 302–320.

Weske, M., 2007. Business Process Management: Concepts, Languages, Architectures. Springer-Verlag, Leipzig, Berlin Heidelberg, Germany, p. 368.

Wolf, F.M., 1986. Meta-analysis: Quantitative Methods for Research Synthesis. Sage Publications.

Zou, Y., et al.,2004. Model-driven business process recovery. In: Proceedings of the 11th Working Conference on Reverse Engineering (WCRE 2004). IEEE Computer Society, pp. 224–233.

Ismael Caballero is assistant professor at the University of Castilla-La Mancha and belongs to the Alarcos Research Group at the UCLM. He holds the PhD degree in Computer Science from the University of Castilla-La Mancha. His research interests include software and data quality, database design and software development based on quality. Contact him at Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; Ismael.Caballero@uclm.es.

Ignacio García-Rodriguez de Guzmán is assistant professor at the University of Castilla-La Mancha and belongs to the Alarcos Research Group at the UCLM. He holds the PhD degree in Computer Science from the University of Castilla-La Mancha. His research interests include software maintenance, software modernization and service-oriented architecture. Contact him at Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; Ignacio.GRodriguez@uclm.es.

Mario Piattini is full professor at the UCLM. His research interests include software quality, metrics and maintenance. He holds the PhD degree in Computer Science from the Technical University of Madrid, and leads the Alarcos Research Group at the Universidad de Castilla-La Mancha. He is CISA, CISM e CGEIT by ISACA. Contact him at Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; Mario.Piattini@uclm.es.

Ricardo Pérez-Castillo holds the MSc degree in Computer Science from the University of Castilla-La Mancha, and he is currently a PhD student in Computer Science. He Works at the Instituto de Tecnologías y Sistemas de Información (ITSI) at the University of Castilla-La Mancha. His research intererests include architecture-driven modernization, model-driven development and business process mining. Contact him at Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; Ricardo.PdelCastillo@uclm.es.