An Empirical Comparison of Static and Dynamic Business Process Mining

Ricardo Pérez-Castillo, Ignacio García-Rodriguez de Guzmán and Mario Piattini
University of Castilla-La Mancha
Paseo de la Universidad 4 13071
Ciudad Real, Spain
+34926295300
{ricardo.pdelcastillo, ignacio.grodriguez, mario.piattini}@uclm.es

Barbara Weber
Technikerstraße 21a, 6020
Innsbruck, Austria
barbara.weber@uibk.ac.at

Ángeles S. Places
Universidade da Coruña
Facultade de Informática, Campus de Elviña s/n, 15071
A Coruña, Spain
asplaces@udc.es

ABSTRACT
Legacy information systems age over time as a consequence of the uncontrolled maintenance and need to be modernized. Process mining allows the discovery of business processes embedded in legacy information systems, which is necessary to preserve the legacy business knowledge, and align them with the new, modernized information systems. There are two main approaches to address the mining of business processes from legacy information systems: (i) the static approach that only considers legacy source code's elements from a syntactical viewpoint; and (ii) the dynamic approach, which also considers information derived by system execution. Unfortunately, there is a lack of empirical evidence facilitating the selection of one of them. This paper provides a formal comparison of the static and dynamic approach through a case study. This study shows that the static approach provides better performance, while the dynamic approach discovers more accurate business processes.

Categories and Subject Descriptors
D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement – *Restructuring, reverse engineering, and reengineering.* D.2.8 [**Software Engineering**]: Metrics – *Performance measures*.

General Terms
Measurement, Performance and Experimentation.

Keywords
Business Process Mining, Software Modernization and Case Study

1. INTRODUCTION
Enterprise information systems age over time as a consequence of the uncontrolled software maintenance becoming Legacy Information Systems (LIS) [26]. When the maintainability of a LIS decreases under acceptable limits, their modernization becomes necessary [24] (i.e. re-implementation of the system with an improved design or a better technology). However, software modernization involves an important risk, i.e., the loss of business knowledge that was progressively embedded in enterprise information systems [12]. This is valuable knowledge since it implicitly represents the organization's business processes, and must therefore be preserved when modernizing the respective information systems.

To deal with this risk and preserve the business knowledge embedded in LISs, the first step is the elicitation of this knowledge. In this sense, a manual business process modeling from scratch by business experts is usually discarded since it is a time-consuming and error-prone choice and does not ensure full knowledge preservation [25]. Instead, business-process mining has become the most powerful and mature approach to discover current business processes considering knowledge of LISs using little manual intervention [3].

There is a large amount of work in the literature about business knowledge preservation and business-process mining. There are mainly two approaches to carry out business-process mining: the static and dynamic approach. On one hand, the static approach considers the source LIS as static entity, i.e., the business knowledge is recovered by statically analyzing the different legacy software artifacts of the LIS. On the other hand, the dynamic approach additionally considers the LIS from a dynamic viewpoint, i.e., it recovers business knowledge derived from the system execution. Regrettably, there are not formal empirical studies to ensure what approach is better in terms of their effectiveness and efficiency. Therefore, the decision of selecting one of these approaches under specific conditions cannot be appropriately taken.

In this paper we present MARBLE [20], a modernization framework to recover business processes from LISs, as well as two process mining techniques within MARBLE based on static and dynamic analysis of source code. The main contribution of this paper is a formal case study involving a real-life author management system to compare the effectiveness and efficiency of these two approaches. Our final goal is to provide a better understanding of strengths and weaknesses of the static and dynamic approaches in order to know when to apply one. The results of the case study show that the static solution provides less

SAC'11, March 21-25, 2011, TaiChung, Taiwan.

effectiveness but a better performance. The dynamic solution, in turn, allows recovering more meaningful processes representing more accurately the organization's current business processes.

The remainder of this paper is organized as follows. Section 2 summarizes related work. Section 3 briefly presents MARBLE, the modernization framework to recover business processes. Section 4 presents the static solution and Section 5 shows the dynamic solution within MARBLE. Section 6 provides the case study to evaluate both solutions. Finally, Section 7 discusses conclusions and future work.

2. RELATED WORK

There are many works related to business-process mining. Several works address business-process mining using static analysis of source code like *Zou et al.* [27] that developed a framework to recover workflows from LISs by applying a set of heuristic transformation rules. Beside source code, other software artifacts are also considered to obtain business processes in a static way, e.g. *Ghose et al.* [8] propose a set of text-based queries in documentation for extracting business knowledge. System databases are other used artifacts, e.g. *Paradauskas et al.* [19] recover business knowledge through the inspection of the data stored in databases.

All these works solely rely on static analysis, which has the disadvantage that a lot of knowledge is lost since it disregards all runtime knowledge. Therefore, alternative solutions based on dynamic analysis have been simultaneously suggested supporting process mining. Dynamic analysis has been applied for a wide set of topics [4]. For instance, *Eisenbarth et al.* [7] present a feature location technique based on dynamic analysis, which gathers the information from a set of scenarios invoking the features. These scenarios are previously defined by domain experts in a manual way. *Cai et al.* [2] propose an approach which combines requirement reacquisition with dynamic analysis. Firstly, a set of use cases is recovered by interviewing the system's users. Secondly, the system is dynamically traced based on these use cases to recover business processes. *Di Francescomarino et al.* [6] recover business processes by dynamically analyzing the Web application GUI-forms which are executed during user's navigation.

Other works addressing the dynamic approach provide process mining techniques that register event logs. Event logs depict the sequence of business process' activities executed, and can be used to discover the current business processes. In this sense, *Günther et al.* [9] provides a generic import framework for obtaining event logs from different kinds of process-aware information systems. In addition, *Ingvaldsen et al.* [10] focus on ERP systems to obtain event logs from the SAP's transaction data logs.

3. MARBLE

Both the static and dynamic solution are framed in MARBLE, a *Modernization Approach for Recovering Business processes from LEgacy systems* [20]. MARBLE is a framework to facilitate business processes mining from LISs based on ADM (Architecture Driven Modernization). ADM is the standard for software modernization defined by the OMG [15], which advocates carrying out reverse engineering processes following the model driven development principles. MARBLE also uses another important standard, KDM (Knowledge Discovery Metamodel) [11], which enables the representation and management of the knowledge extracted by means of reverse engineering from all the different software artifacts of LISs in an integrated way. That legacy knowledge is then gradually transformed into business processes. For this reason, MARBLE defines four kinds of models at four different abstraction levels (see Figure 1). *L0* is the lowest level of abstraction since it represents the LIS in the real world as a set of different software artifacts (e.g. source code, database, documentation, etc). *L1* contains different platform-specific models (PSM) depicting the different software artifacts of the LIS. *L2* integrates all the specific *L1* models into a platform-independent model (PIM), which is represented according to the KDM metamodel. Finally, *L3* depicts the discovered business processes, which are represented according to the BPMN (Business Process Model and Notation) metamodel [17]. In addition, other kind of models can be extracted from the KDM model at L2, for instance, service model, business-rules model, among other similar models.

Moreover, MARBLE defines three model transformations between the four levels (see Figure 1). (i) The L0-to-L1 transformation obtains PSM models from each legacy software artifact using a specific metamodel for each artifact. The traditional reverse engineering techniques such as static analysis, dynamic analysis, program slicing and dicing, formal concept analysis, subsystem decomposition, and so on, can be used to extract the needed knowledge. (ii) The L1-to-L2 transformation consists of a set of model transformations (e.g. implemented using QVT (Query/View/Transformation) [16]) to obtain a KDM model built from the PSM models at L1. (iii) The L2-to-L3 transformation finally obtains the current business process model (see Figure 1). This transformation is based on a set of business patterns, which define the transformation rules between levels L2 and L3. In addition, this last transformation can be supported by business experts who know the organization and can provide additional meaningful knowledge. They can detect inconsistent or incoherent fragments in the preliminary business process models obtained after pattern matching. Thereby, they can refactor the business process models, add manual activities to the preliminary models, and so on.

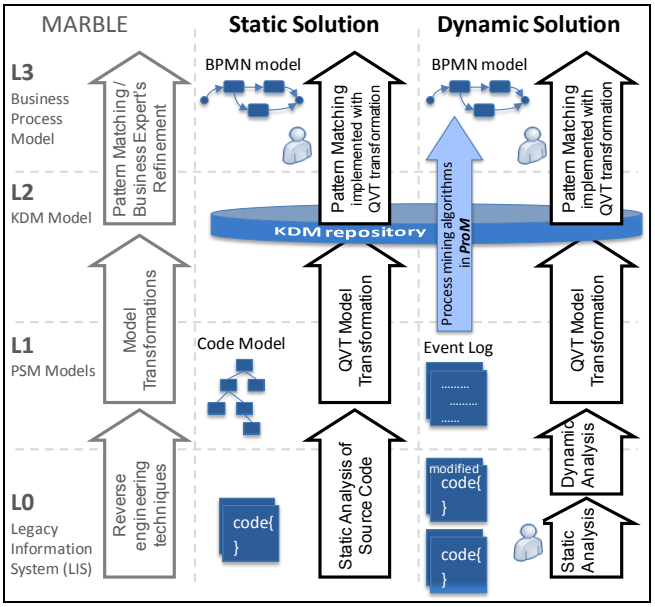

Figure 1. Static and dynamic process mining techniques

4. STATIC SOLUTION

The static solution, framed in MARBLE, considers: (i) legacy source code as the key software artifact at L0, and static analysis as the reverse engineering technique to extract the embedded knowledge at L0 and represent it at L1. Static analysis of source code consists of the sequential, syntactic inspection of the all source code files.

This solution is supported by a tool that was especially developed for this purpose. The tool has a module to analyze the source code file (Java files in particular) and builds an abstract syntax tree of the source code, i.e. a code model at level L1(see Figure 1). This parser was build using *JavaCC* [18], an open source parser generator for Java. The advantage of the static approach is that a syntactic parser for analyzing the source code is easy and inexpensive to build. Moreover, the tool implements a QVT model transformation to transform the code model into a KDM model at L2. The KDM model provides a standard inventory of all software artifacts, thus it can be used for any process mining technique or any other software modernization activity.

The static solution (see Figure 1) also provides a set of business patterns [21], thus when a specific structure is detected in the KDM model at level L2, each pattern indicates what elements should be built and how they are interrelated in the business process model at level L3. The set of patterns are divided into three categories. (A) *Structural patterns* deal with the built elements and their combinations. There are four patterns that transform: (i) packages, compilation units or other aggregation units (e.g. Java classes or interfaces) into business process diagrams; (ii) methods into tasks; (iii) calls between methods into sequence flows between tasks; and finally (iv) conditional branching (e.g. *if-then-else* or *switch* statements) into exclusive gateways that branch the sequence flow. (B) *Data patterns* deal with data objects and how these objects are related to other elements. There are two data patterns transforming: (i) program variables read for a method into a data object with an association to the respective task; and (ii) program's variables written into data objects with an association from the task. (C) Finally, *Event patterns* build all the elements involved in the event management. There are three patterns, which transform: (i) the start method into a start event and sequence flow to the respective task; (ii) end tasks into sequence flows from the tasks to an end event; and (iii) conditional calls into sequence flows with an intermediate conditional event.

The pattern recognition and the generation of business process models is also implemented in the developed tool by means of QVT transformations [22]. In addition, the discovered business processes can be refined by business experts through the tool, since it provides graphical model editors for both KDM and business process models.

5. DYNAMIC SOLUTION

The dynamic solution is also framed in MARBLE (see Figure 1), although in this case, knowledge derived by system execution is considered. Thereby, the reverse engineering technique considered in the L0-to-L1 transformation combines static analysis and dynamic analysis. Firstly, the static analysis inspects and modifies the original legacy source code to enable the registration of event logs as explained below. After that, the dynamic analysis is registering the even log during system execution.

The static pre-analysis injects specific statements in certain places of the source code (as explained below) to register the execution events in a log when these statements are reached (see Figure 1). Each event registered in the log specifies the execution of an underlying business task supported by a certain piece of source code. Thereby, to inject appropriately the special statements in the source code there are five key challenges that must be addressed according to [23]: (i) process definitions are implicit described in legacy code and, thus, it is not obvious which events should be recorded in the event log; (ii) the granularity of callable units of an information system and activities of a business process often differs; (iii) legacy code not only contains business activities, but also technical aspects which have to be discarded when mining a business process; (iv) since traditional systems do not explicitly define processes, it has to be established when a process starts and ends; (v) finally, due to the missing process-awareness, it is not obvious how business activities and process instances should be correlated.

The proposed solution partially solves some challenges with specific information provided by business experts and system analysts. Firstly, business experts establish the start and end business activities of the business processes to be discovered, which are needed to define the process scope. In parallel, system analysts examine the legacy source code and filter the directories, files or set of methods supporting business activities. This information is necessary to reduce potential noise in the event log due to technical source code. Secondly, system analysts do the mapping between start/end business activities and the methods supporting them. In addition, system analysts define the *correlation data set* which uniquely identifies a process instance. It is needed to relate each executed business activity in a process instance, since according to the fifth challenge [23] business process is typically not only executed once, but multiple instances are executed concurrently. If a particular business activity is executed (i.e., a method is invoked), this particular event has to be correctly linked to one of the running process instances by means of the *correlation data set*. System analysts define what parameter(s) in each method (a candidate business task) contains the correlation data set. During system execution, the parameter's data is used to put the task, created from the method, in the process instance that groups tasks with that correlation data. For instance, in a healthcare information system, the correlation data set could be the patient information, since it determines different instances of a business process, e.g. the process *'patient admission'*.

The manual intervention of both business experts and system analysts might appear as a time-consuming task. However, it is a task supported by an *ad hoc* tool that was developed to support the dynamic solution, which enables the quick and easy collection of all this information. The static pre-analysis is the stage that spends more time compared with the dynamic analysis stage, however it must be done only once, and the dynamic analysis to register event logs can be done repeatedly without additional manual effort.

After manual intervention, the syntactic inspection of the source code is automatically carried out by means of a parser, which analyzes and injects on the fly the special statements. During the static pre-analysis, the source code is broken down into methods, which are considered as candidate business tasks, and then, the parser only modifies the non-technical methods filtered by system analysts. In addition, fine-grained methods (e.g., *setter*, *getter*,

constructor, *toString* and *equals* methods) are automatically discarded to reduce the noise of the event log (i.e., they are not used to create events). Finally, in each filtered method, two statements are injected at the beginning and end of each one. The first statement represents an event with a *start* event type, and the second one represents the *complete* event for the same business task. Moreover, the correlation data set defined for the method as well as information whether or not the method represents a start or end task are included in the statements.

After static analysis, the dynamic analysis is carried out (see Figure 1). When the modified code is executed, the injected statements will invoke a function writing the respective event in the event log. The event log can be used then to discover the business processes taking the system execution information into account. In the same manner than the static solution, the dynamic solution transforms the event log at L1 into a KDM model at L2, and then this model into business process models at L3. However, in this case we do not develop additional tools to discover business processes from event logs, since there exists *ProM* [14], the world-leading tool in the area of process mining, which supports a lot of control-flow techniques and algorithms to discover processes from event logs (see Figure 1).

6. CASE STUDY

This section provides an empirical study to compare both the static and dynamic solution in terms of their effectiveness and efficiency. The case study is carried out following the formal protocol for planning, conducting and reporting case studies proposed by *Brereton et al* [1], improving the rigor and validity of the study. The following sections present the stages of the protocol in detail: background, design, case selection, case study procedure, data collection, analysis and interpretation, and validity evaluation.

6.1 Background

Firstly, the previous research on the topic must be identified. The related work presented in Section 2 discusses other proposals for recovering business knowledge from LISs. Particularly, our proposal focuses on MARBLE, an ADM-based framework to obtain business processes. The *object of study* are the two process mining solutions framed in MARBLE, respectively following the static and dynamic approach. The *purpose of this study* is the evaluation of specific properties of the proposed solutions related to their effectiveness and efficiency, and the comparison between them.

Taking into account the object and purpose of the study, two main research questions (MQ) can be defined (see Table 1). On one hand, *MQ1* checks if the solutions can effectively discover business processes, i.e., evaluates whether the processes recovered from the LIS represent the business behavior of the organization that owns the information system. In addition, *MQ1* is divided into three additional research questions (AQ) to be answered: (i) *AQ1* evaluates whether the discovered business processes comprise all the elements of the organization's current business processes; and (ii) *AQ2* checks whether the mined processes contain any elements which do not belong to the current business processes. Moreover, *MQ2* is related to the efficiency of the proposed solutions to evaluate if it might be used with larger information systems. MQ2 considers the additional question *AQ3* to evaluate the time spent on the static and dynamic analysis of source code with regard to the size of the system.

Table 1. Case study research questions

Id	Research Question
MQ1	Can the solution effectively mine business processes from LISs?
AQ1	Can the solution discover all the relevant elements of the embedded business processes?
AQ2	Can the solution discover the embedded business processes without the discovery of unnecessary elements?
MQ2	Is the solution efficient to be scaled to any LIS?
AQ3	Is the performance time linear with regard to the size of the LIS?

6.2 Design

The case study focuses on a sole LIS (a single case). Due to the features of each solution, the design of the study is different in each case. On one hand, the study execution concerning the static solution follows an *embedded* design since it is applied considering each source code package of the system as an analysis unit. The static solution must consider a minimal analysis unit to progressively transform it into a business process model at L3. After that, business process models can be joined or split in the manual post-intervention to fit to the reality of the organization. On the other hand, the study execution of the dynamic solution follows a *holistic* design since it is applied to the case as a whole, and does not consider several analysis subunits.

After the application of the proposed solutions obtaining the current business processes, which are considered as the independent variable, they are analyzed to answer the research questions (see Table 1). In order to quantitatively answer the questions some measures are established as dependent variables. The study use the *precision* and *recall* measures [5] to answer questions *AQ1* and *AQ2* respectively. These measures are used because precision can be seen as a measure of exactness or fidelity, whereas recall is a measure of completeness. *Precision* (1) represents the amount of relevant recovered tasks within the set of recovered task in a business process model. *Recall* (2) represents the amount of relevant recovered tasks of the total of relevant tasks (recovered and non-recovered) that depict the whole business operation of the organization. Business expert opinion is used in both measures to determine if a task is or is not relevant (i.e. the task faithfully represents the business operation or behavior of the organization in the real world). Although *precision* and *recall* are adequate, there is an inverse relationship between them. As a consequence, extracting conclusions to answer *MQ1* with an isolated evaluation of these measures is very difficult. For this reason, these measures are usually combined into a single measure known as *F-measure* (3), which consists of a weighted harmonic mean of both measures.

$$PRECISION = \frac{|\{relevant\ tasks\} \cap \{recovered\ tasks\}|}{|\{recovered\ tasks\}|} \quad (1)$$

$$RECALL = \frac{|\{relevant\ tasks\} \cap \{recovered\ tasks\}|}{|\{relevant\ tasks\}|} \quad (2)$$

$$Fmeasure = \frac{2 \cdot PRECISION \cdot RECALL}{\text{PRECISION} + \text{RECALL}} \quad (3)$$

Moreover, to evaluate the *MQ2* and its sub-questions *AQ3* the study uses the total time spent by each solution. The L0-to-L1 transformation characterizes each solution, since the computational cost of remaining transformations are equals. Thus, the time spent on this transformation is the time evaluated. The static solution's time (4) for *'n'* executions is the time spent on the static analysis of source code multiplied by *'n'*. The dynamic solution's time (5) consists of a constant time spent on manual intervention and static pre-analysis as well as the time spent on the dynamic analysis multiplied by the *'n'* times executed. This dynamic analysis time represents the performance penalty due to the modified source code.

$$T_{\{S,n\}} = n \cdot T_{\{Static\ Analysis\}} \quad (4)$$

$$T_{\{D,n\}} = T_{\{Manual\ Intervention\}} + T_{\{Static\ Analysis\}} + n \cdot T_{\{Dynamic\ Analysis\}} \quad (5)$$

6.3 Case Selection

Case selection is a key stage in case study planning. Table 2 presents the four criteria to select a good and suitable case to be studied. After the evaluation of several available systems according to the criteria, the information system selected for study was *"AELG-members"*, which supports the administration of an organization of Spanish authors. The system automates several services offered by the organization, including, among others, author registration, cancelation of memberships and payment of fees. For this reason, the system meets *C1*. Moreover, the first release of *AELG-members* was moved to the production stage 2 years ago, and it has had three medium modifications and a large modification (versions 1.1, 2.0, 2.1, and 2.2), thus C2 is satisfied. From a technological point of view, the system is a Java application meeting C4, and has 23.5 KLOC (thousands of lines of source code) ensuring C3.

Table 2. Criteria for case selection

Id	Criterion for case selection
C1	It must be an enterprise system
C2	It must be a LIS
C3	It must be of a size not less than 10 KLOC
C4	It must be a Java-based system

6.4 Case Study Procedure

After design and selection of the case study, the study's execution procedure must also be planned. The execution is supported by the tools developed to support both solutions. The case study procedure defines the followings steps. (i) After discussion meetings between staff of the candidate organizations and researchers, the LIS is selected according to the selection criteria. In addition, the business expert and system analyst participating in the study are appointed in this step. (ii) The legacy source code is analyzed (and modified) by applying both static and dynamic solutions through the respective tool. (iii) In the case of static solution a code model is obtained at L1, and in the dynamic solution, the modified system is implanted in the organizational environment and it is used by some of the usual users in the organization for two months to generate a meaningful event log at L1. (iv) After that, the code model and event log obtained through static and dynamic solutions respectively are used to discover the current business processes according to the proposed solutions. (v) Business experts then fit the preliminary business processes with the reality of the organization to evaluate the *precision* and *recall* measures. (vi) All key information related to the generation of the business processes (steps ii-iv), as well as the business expert intervention (step v), is collected according to the data collection plan (see Subsection 6.5). (vii) The data collected in the previous step is analyzed and interpreted to draw conclusions in order to answer the research questions (see Subsection 6.6). Finally, the case study is reported and feedback is given to the organization and research community.

6.5 Data Collection

The data collection plan is defined before starting the execution of the case study. Firstly, data concerning the static analysis and the business expert's configuration to obtain the event log are recorded. The time spent on process mining with each solution is also annotated. After discovery of the preliminary business process models using both solutions, business experts modify them and obtain the three final models which are then used for evaluating the proposed measures. Table 3 shows (i) the number of recovered tasks (before manual intervention); (ii) the number of recovered relevant tasks (i.e., the number of tasks that the business experts mark as correct); (iii) the number of recovered non-relevant tasks (i.e., tasks removed from the business process since they do not represent a business activity); (iv) the precision, (v) the recall and (vi) the F-measure values for each final business process; (vii) the total number of relevant tasks (recovered or non-recovered); and finally the total time (in milliseconds) spent on the L0-toL1 transformation.

6.6 Analysis and Interpretation

After the data has been collected, it is analyzed to obtain the evidence chains from the data to answer the research questions and draw conclusions. The application of both solutions obtained three business processes: (i) *Categories Management*; (ii) *Author Management*; and (iii) *Reporting*.

Table 3. Final business process data

	Business Process Model	# Rec. tasks	# Rec. relev. tasks	# Rec. non-relev. tasks	# Non-rec. relev. tasks	Precision	Recall	F-Measure	# Total relev. tasks	$T_{\{Static\ Analysis\}}$
Static	Categories Mgmt.	69	40	29	12	0.580	0.769	0.661	52	22
	Author Mgmt.	141	71	70	9	0.504	0.888	0.643	80	35
	Reporting	36	16	20	8	0.444	0.667	0.533	24	14
	Mean	82.0	42.3	39.7	9.7	0.509	0.774	0.612	52	23.7
	Std. Deviation	53.7	27.6	26.6	2.1	0.07	0.11	0.07	28	10.6
Dynamic	Categories Mgmt.	14	10	4	4	0.714	0.714	0.714	14	50
	Author Mgmt.	27	15	12	2	0.556	0.882	0.682	17	65
	Reporting	6	5	1	2	0.833	0.714	0.769	7	12
	Mean	15.7	10.0	5.7	2.7	0.701	0.770	0.722	13	42.3
	Std. Deviation	10.6	5.0	5.7	1.2	0.14	0.10	0.04	5	27.3

Firstly, to answer question *MQ1*, questions *AQ1* and *AQ2* must be evaluated (see Figure 2). The mean of the precision and recall measures was p=0.509, r=0.774 for the static solution and p=0.701, r=0.770 for the dynamic solution (see Table 3). Recall

values are similar, although the dynamic solution provides a slightly better value than the static one. In both cases, a large amount of non-relevant tasks has been recovered. In most cases, respective tasks did not represent business activities, but rather technical source code (e.g. auxiliary methods). In addition, several tasks have been classified as non-relevant due to their inappropriate level of granularity (e.g., their business knowledge is already included in other larger relevant tasks). Precision values, in turn, are much higher for the dynamic solution than the static one. This is due to two main reasons. Firstly, the dynamic solution can ignore, for instance, the dead parts of source code since it considers the system execution information. Secondly, the dynamic solution follows an expert-driven approach. While the static solution automatically considers each source code package as a candidate business process model, the process definition of the dynamic solution is partially supported by information provided by business experts, which helps to discard technical source code or mitigate the granularity problem.

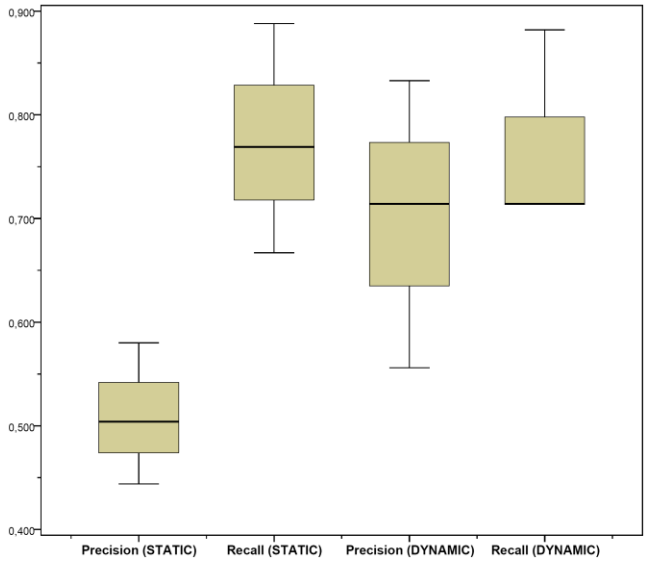

Figure 2. Precision/recall box plot for each solution

In both cases a high recall value contrasts with a lower precision value, which means that the number of non-relevant tasks is very high with respect to the recovered tasks. This means that both solutions obtain large business processes which need to be greatly reduced by removing several non-relevant tasks.

The results obtained are usual since there is an inverse relationship between precision and recall measures. This means that the proposed solutions might reduce its recall score by recovering fewer tasks, at the cost of reducing the number of non-relevant recovered tasks, i.e., increasing the precision score. These supposed values are more desirable, since the precision and recall would be more balanced. In this sense, the F-measure (see Table 3), which summarizes both measures, shows that the dynamic solutions with 0.722 is more effective than the static solution with 0.612. The precision and recall values of the dynamic solution are not only more balanced, but in addition, these values are individually better (see Figure 2).

In any case, to answer *AQ1* and *AQ2*, and in consequence *MQ1*, the obtained values were additionally compared with reference values from other experiences with model recovery in literature [13], which report precision and recall values close to 50%. The values obtained by applying both approaches were above 0.5, the benchmark value. Thereby, *AQ1* and *AQ2*, and as a consequence *MQ1,* can be answered positively, i.e., both solutions can recover business processes form LISs with a sufficient effectiveness level. However, the dynamic solution presents better results from the effectiveness point of view, since precision and recall indicators are better than the static solution.

Furthermore, question *AQ3* must be answered to evaluate both solutions from an efficiency point of view. The average time $T_{\{Static\ Analysis\}}$ was 23.7 ms for the static solution and 42.3 ms for the dynamic solution (see Table 3). In addition, the dynamic solution has two additional time penalties. Firstly, $T_{\{Dynamic\ Analysis\}}$ as a tiny performance penalty due to the injected sentences execution. This penalty is constant with respect to the system size, since it only affects the response time of each system's service or functionality in particular. For this reason $T_{\{Dynamic\ Analysis\}}$ can be considered as a negligible value. Secondly, the time spent on the manual intervention of business experts and system analysts to provide the needed information, which was $T_{\{Manual\ Intervention\}} = 90$ minutes. As a consequence, the final time values were (on average) $T_{\{S,1\}} = 0.024$ seconds for the static solution, and $T_{\{D,1\}} = 5400+0.042+0$ seconds for the dynamic solution. The dynamic solution's bottleneck is obviously the manual intervention for a sole dynamic analysis (i.e. n=1). However, the additional advantage of the dynamic approach is that several iterations are possible further improving the quality of the event log. In the subsequent iterations, the manual intervention effort is progressively reduced until it is not needed, thus the $T_{\{S,n\}}$ and $T_{\{D,n\}}$ tend to be equals for large *'n'* values.

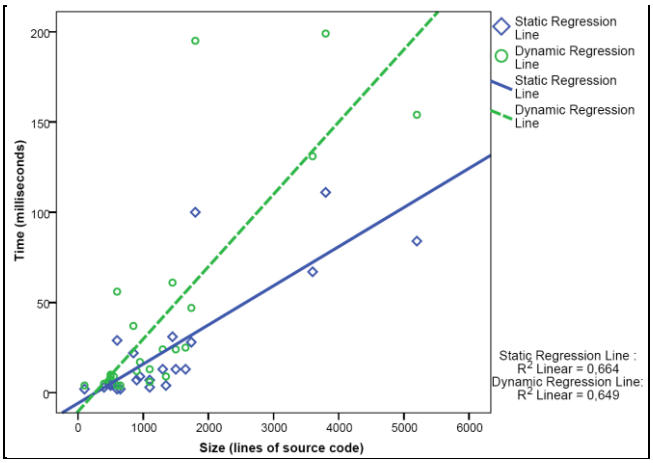

Figure 3. Linear regression model to evaluate the scalability

In any case, the time values of both solutions seem feasible for the selected case with 23.5 KLOC. However, the scalability of the technique must be evaluated. For this purpose, a linear regression model was considered with time as dependent variable and size of each java package in lines of source code as independent variable. Figure 3 presents the scatter chart of size/time showing the regression lines which present positive linear relationships between the package size and the analysis time. The time variable in the regression model considers the partial time $T_{\{Static\ Analysis\}}$ for each java package. To compare time values through the regression model, it not considers $T_{\{Manual\ Intervention\}}$ for the dynamic solution

since this time only represents an increase in the vertical axis while the slope of the regression line is not affected. To have a better graphical comparison, this increase is not represented in the chart (see Figure 3). The correlation coefficients R^2 of the regression model (i.e. the degree to which the real values of the dependent variable are close to the predicted values) were 0.66 and 0.65 for static and dynamic solutions respectively. These values are high for a positive linear relationship with R^2 values between 0 and 1. The proposed linear regression models are therefore suitable to explain the results of both solutions. There is not a quadratic or exponential relationship between time and size, thus the expected increase in time for larger systems will consequently be linear, and the time will be assumable. Question *AQ3*, and consequently *MQ2*, can therefore be answered as true for both solutions.

Moreover, both static and dynamic solutions are compared. The regression line of static solution has a lower slope and the average time for each process is also lower than the value of the dynamic solution. In addition, the dynamic solution has the added time $T_{\{Manual\ Intervention\}}$. Therefore, the dynamic solution performance is worse than the static one in terms of the efficiency.

6.7 Validity Evaluation

After the analysis and interpretation of the results, the validity of the study must be evaluated, i.e. if the results are true and not biased for the whole population for which we want to generalize the results. There are mainly three types of validity: internal, construct and external. This section shows the threats to the validity and the list of actions to mitigate them.

Firstly, the *internal validity* is threatened by two main factors. The first threat is related to the tools used to support both solutions, since the measures might be different if the business processes are obtained using another tool. To mitigate this threat, the study could be replicated using different tools and the obtained results could be compared. The second threat is that the results concerning the precision and recall measures could be also biased due to the manual intervention by business experts, since it can represent a subjective point of view. This threat is difficult to eradicate, however different business experts teams could be considered to have different viewpoints.

According to the *construct validity*, the proposed measures were adequate to measure the variables and answer the research questions appropriately. The precision and recall measures were reused from the information retrieval field, where these measures have an adequate maturity level. In addition, these measures allow us to check whether the business processes obtained accurately represent (or not) the business behavior of the organization. Nevertheless, the benchmark value of 0.5 taken from literature to compare the obtained results may be quite relative. Unfortunately, there exit not enough benchmark values for these metrics in the process mining field, thus this threat is not easy to mitigate at the moment. Another threat to the construct validity is the duration of the dynamic analysis (two months in this study), since there is no metric to know how much time is necessary to obtain meaningful event logs considering all possible scenarios. Maybe, if the log is collected for more than two months, the precision and recall value could be better. To mitigate this threat the study could be replicated considering different dynamic analysis durations.

Finally, *external validity* is concerned with the generalization of the results to a whole population. The obtained results could be generalized to LISs. Nevertheless, the specific program language of the selected case is a threat that should be noted, sine the results can be strictly extended to those LISs based on Java language. Anyway, the expected results for other kinds of object-oriented systems could be quite similar to these results. The study should be replicated and compared using information systems based on other different platforms to mitigate this threat.

7. CONCLUSION

This paper has presented an empirical study to compare two different but related process mining solutions following the static and dynamic approaches. Both solutions are framed in MARBLE, an ADM-based framework that facilitates the modernization of LISs by means of the discovery of the current business processes embedded in this kind of systems. On one hand, we present the static solution, a process mining technique that carries out the syntactic analysis of legacy source code to extract meaningful business knowledge to rebuild the current business processes. On the other hand, the dynamic solution takes the information about the system execution into account. Firstly, the dynamic solution statically analyzes legacy source code and injects special sentences in certain places within source code. Secondly, the instrumentalized source code is able to write event logs during its execution. The event logs are then used to discover the current business processes.

Process mining from LISs is a common and real problem that companies and academics have been trying to solve for many years. The static and dynamic approaches are the two main choices to discover the current business processes. The main difference between both approaches is whether or not the process mining proposals consider the system execution information. Despite all the solutions proposed in literature, there exist no empirical studies showing evidences of what approach is better, and under what conditions.

The study result shows that the performance of the static solution is better than for the dynamic solution. However, the dynamic solution provides more accurate business processes than the static one. Therefore, due to the fact that both solutions might be used with larger LISs with a linear time increase, the dynamic approach is more suitable than the static approach.

The work in progress is dealing with replications of this study with more LISs considering different conditions like other platforms, other tools supporting the solutions, different dynamic analysis durations, and so on. The ultimate objective is to provide more comparisons such that the results can be generalized to any process mining scenario.

Moreover, future work will address the combination of static and dynamic approaches. The static analysis, and specially the data patterns, might be very helpful for tracing correlation data sets during dynamic analysis to correlate each activity in the correct business process instance.

ACKNOWLEDGMENTS

This work was supported by the FPU Spanish Program funded by MICINN; by the R+D projects funded by JCCM: ALTAMIRA (PII2I09-0106-2463) and PRALIN (PAC08-0121-1374); as well as the PEGASO/MAGO project (TIN2009-13718-C02-01) funded by MICINN and FEDER. In addition, this work was supported by the Quality Engineering group at the University of Innsbruck.

8. REFERENCES

[1] Brereton, P., B. Kitchenham, D. Budgen, and Z. Li. "Using a protocol template for case study planning". in Evaluation and Assessment in Software Engineering (EASE'08). 2008. Bari, Italia p. 1-8.

[2] Cai, Z., X. Yang, and W. Wang. "Business Process Recovery for System Maintenance - An Empirical Approach". in 25 th International Conference on Software Maintenance (ICSM'09). 2009. Edmonton, Canada: IEEE CS p. 399-402.

[3] Castellanos, M., K.A.d. Medeiros, J. Mendling, B. Weber, and A.J.M.M. Weitjers, Business Process Intelligence, in Handbook of Research on Business Process Modeling, J. J. Cardoso and W.M.P. van der Aalst, Editors. 2009, Idea Group Inc. p. 456-480.

[4] Cornelissen, B., A. Zaidman, A.v. Deursen, L. Moonen, and R. Koschke, "A Systematic Survey of Program Comprehension through Dynamic Analysis". IEEE Trans. Softw. Eng., 2009. 35(5): p. 684-702.

[5] Davis, J. and M. Goadrich. "The relationship between Precision-Recall and ROC curves". in Proceedings of the 23rd international conference on Machine learning. 2006. Pittsburgh, Pennsylvania: ACM p. 233-240.

[6] Di Francescomarino, C., A. Marchetto, and P. Tonella. "Reverse Engineering of Business Processes exposed as Web Applications". in 13th European Conference on Software Maintenance and Reengineering (CSMR'09). 2009. Fraunhofer IESE, Kaiserslautern, Germany: IEEE Computer Society p. 139-148.

[7] Eisenbarth, T., R. Koschke, and D. Simon, "Locating Features in Source Code". IEEE Trans. Softw. Eng., 2003. 29(3): p. 210-224.

[8] Ghose, A., G. Koliadis, and A. Chueng. "Process Discovery from Model and Text Artefacts". in IEEE Congress on Services (Services'07). 2007 p. 167-174.

[9] Günther, C.W. and W.M.P. van der Aalst, "A Generic Import Framework for Process Event Logs". Business Process Intelligence Workshop (BPI'06), 2007. LNCS 4103: p. 81-92.

[10] Ingvaldsen, J.E. and J.A. Gulla, "Preprocessing Support for Large Scale Process Mining of SAP Transactions". Business Process Intelligence Workshop (BPI'07) 2008. LNCS 4928: p. 30-41.

[11] ISO/IEC, ISO/IEC DIS 19506. Knowledge Discovery Meta-model (KDM), v1.1 (Architecture-Driven Modernization). http://www.iso.org/iso/catalogue_detail.htm?csnumber=32625. 2009, ISO/IEC. p. 302.

[12] Koskinen, J., J. Ahonen, H. Lintinen, H. Sivula, and T. Tilus, Estimation of the Business Value of Software Modernizations. 2004, Information Technology Research Institute. University of Jyväskylä.

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

[14] Medeiros, A.K., A.J. Weijters, and W.M. Aalst, "Genetic process mining: an experimental evaluation". Data Min. Knowl. Discov., 2007. 14(2): p. 245-304.

[15] OMG. ADM Task Force by OMG. 2007 9/06/2009 [cited 2008 15/06/2009]; Available from: http://www.omg.org/.

[16] OMG, QVT. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification. http://www.omg.org/spec/QVT/1.0/PDF. 2008, OMG.

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

[18] Open Source Initiative, JavaCC 4.2. A parser/scanner generator for java. https://javacc.dev.java.net/. 2009.

[19] Paradauskas, B. and A. Laurikaitis, "Business Knowledge Extraction from Legacy Information Systems". Journal of Information Technology and Control, 2006. 35(3): p. 214-221.

[20] Pérez-Castillo, R., I. García-Rodríguez de Guzmán, O. Ávila-García, and M. Piattini. "MARBLE: A Modernization Approach for Recovering Business Processes from Legacy Systems". in International Workshop on Reverse Engineering Models from Software Artifacts (REM'09). 2009. Lille, France: Simula Research Laboratory Reports p. 17-20.

[21] Pérez-Castillo, R., I. García-Rodríguez de Guzmán, O. Ávila-García, and M. Piattini. "Business Process Patterns for Software Archeology". in 25th Annual ACM Symposium on Applied Computing (SAC'10). 2010. Sierre, Switzerland: ACM p. 165-166.

[22] Pérez-Castillo, R., I. García-Rodríguez de Guzmán, and M. Piattini. "Implementing Business Process Recovery Patterns through QVT Transformations". in International Conference on Model Transformation (ICMT'10). 2010. Málaga, Spain: Springer-Verlag p. 168-183.

[23] Pérez-Castillo, R., B. Weber, I. García Rodríguez de Guzmán, and M. Piattini, "Toward Obtaining Event Logs from Legacy Code". Business Process Management Workshops (BPI'10), 2010: p. In Press.

[24] Ulrich, W.M. and P.H. Newcomb, Information Systems Transformation. Architecture Driven Modernization Case Studies. 2010, Burlington, MA: Morgan Kauffman.

[25] van der Aalst, W.M.P., B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters, "Workflow mining: a survey of issues and approaches". Data Knowl. Eng., 2003. 47(2): p. 237-267.

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

[27] Zou, Y. and M. Hung. "An Approach for Extracting Workflows from E-Commerce Applications". in Proceedings of the Fourteenth International Conference on Program Comprehension. 2006: IEEE Computer Society p. 127-136.