Assessing the Best-Order for Business Process Model Refactoring

María Fernández-Ropero, Ricardo Pérez-Castillo, José A. Cruz-Lemus and Mario Piattini
Instituto de Tecnologías y Sistemas de la Información, University of Castilla-La Mancha
Paseo de la Universidad 4, 13071
Ciudad Real, Spain
+34 926295300 Ext. 96697
{MariaS.Fernandez, Ricardo.PdelCastillo, JoseAntonio.Cruz, Mario.Piattini}@uclm.es

ABSTRACT
Quality assurance is one of the most critical activities in business process models which are obtained by reverse engineering, e.g., from existing information systems. Companies must deal with several quality faults in business process models such as irrelevant elements, fine-grain granularity or incompleteness, which affect understandability and modifiability of business process models. Hence, business process refactoring techniques are often used to improve these features, which change the internal structure of business process models while its external behavior is preserved. Unfortunately, different refactoring operators do not fulfill commutative property among them. For this reason, this paper addresses the challenge of establishing the best order in which to apply all the different refactoring operators and, therefore, to achieve the highest quality improvement. The research methodology consists of conducting a real-life case study to assess the influence of the refactoring operator's order in the understandability and modifiability of business process models. The case study demonstrates that there is a clear influence in these quality features in terms of the size and separability of the business process models under study.

Categories and Subject Descriptors
D.2.7 [**Software Engineering]:** Distribution, Maintenance, and Enhancement – *restructuring, reverse engineering, and reengineering*. D.2.8 [Software Engineering]: Metrics – *process metrics*. I.5.1 [**Pattern Recognition**]: Models – *structural*.

General Terms
Measurement, Performance, Experimentation.

Keywords
Business process model; refactoring; understandability; modifiability; case study.

1. INTRODUCTION
Business processes models define the sequence of coordinated activities carried out by companies to archive their common business goal [1]. Business processes models are considered one of the most important assets for companies. An appropriate management of business process models helps companies to quickly adapt their business goals and structures to environmental changes so that they can maintain or even improve their competitiveness [2]. As a consequence, companies are currently demanding mechanisms to ensure adequate quality levels of their business processes in order to accomplish an adequate business process management. Quality assurance of business process models usually focuses on understandability and modifiability [3], since these proved to be two of the most influent quality features concerning business process management.

Quality assurance is even more critical in business process models that were obtained by reverse engineering, e.g., from existing information systems [4]. Business process mining techniques are often used by companies without ever having done their own business process modeling or with outdated and missing business process models. Business process models retrieved by reverse engineering can undergo a semantic loss due to the abstraction increase, and therefore, these models will probably have inadequate quality degrees.

A suitable solution to improve the quality degree of reverse engineered business process models is applying refactoring techniques [5, 6]. Business process model refactoring changes the internal structure of business process models without altering its external behavior. Business process model refactoring makes it possible to improve understandability and modifiability.

Most business process model refactoring techniques consist of recognizing refactoring opportunities and then these techniques apply different refactoring operators [5, 7, 8]. Refactoring operators replace some process model fragments for semantically equivalent ones. Unfortunately, these refactoring operators do not fulfill commutative property. Hence, one of the most critical challenges is to figure out the best order in which to apply all the different operators to maximize the quality improvement. Refactoring approaches proposed in literature are mainly two: (1) to arbitrarily determine an arbitrary order or (2) to manually define an order by business experts. None of these approaches ensure the best gain of understandability and modifiability.

This paper conducts a case study with which to determine the best order of refactoring operators. The case study firstly retrieved business process models by reverse engineering of a real-life information system. After that, all the business process models were refactored by means of IBUPROFEN, a framework that defines a set of operators [3]. Refactoring operators were executed in six different orders according to their three categories (i.e., irrelevant elements reduction, fine-grain granularity reduction and completeness). The main implication of this work is that the order of refactoring operators affects the understandability and modifiability of business process models.

The remaining of the paper is organized as follows: Section 2 summarizes the background of this work. Section 3 introduces the IBUPROFEN framework and its refactoring operators. Section 4 presents the case study conducted to determine the effect of the order of refactoring operators to the quality degree. Finally, Section 5 discusses conclusions and future work.

2. RELATED WORK

Business process management has become a valuable activity for managing organizations from an operational perspective. *Dijkman et al.* [6] provide various techniques for improving their management as merging, mining, refactoring, re-use, among other. Particularly, refactoring has been used for several authors in literature for improving the quality degree of business process models. For example, *Weber et al.* [5] collect a catalogue of process model *smells* for identifying refactoring opportunities and provide a set of behavior-preserving techniques for refactoring to avoid redundancies and increases in the complexity of the model. Similarly, *Dijkman et al.* [7] show a development of a technique based on metrics to detect refactoring opportunities and *La Rosa et al.* [8] identify patterns to reduce the model complexity through compacting, compositing, merging, amoung other. *Leopold et al.* [9] , for their part, is focus on refactoring of activity labels in a business process model following a verb-object style.

Concerning the order of application of refactoring operators these authors opt for using the expert decision, blocking their automation, or indiscriminately without regard to any particular order.

Other authors such as *Gambini et al.* [10] propose the automation of business process models refactoring through a technique for automatically fixing the scenarios using Petri nets but the order of application is not mentioned.

Unfortunately, none of these works attempt to define best-order execution of the refactoring operators. For this reason, this paper proposes a set of refactoring operators and tries to figure out the best choice between a number of alternative orders.

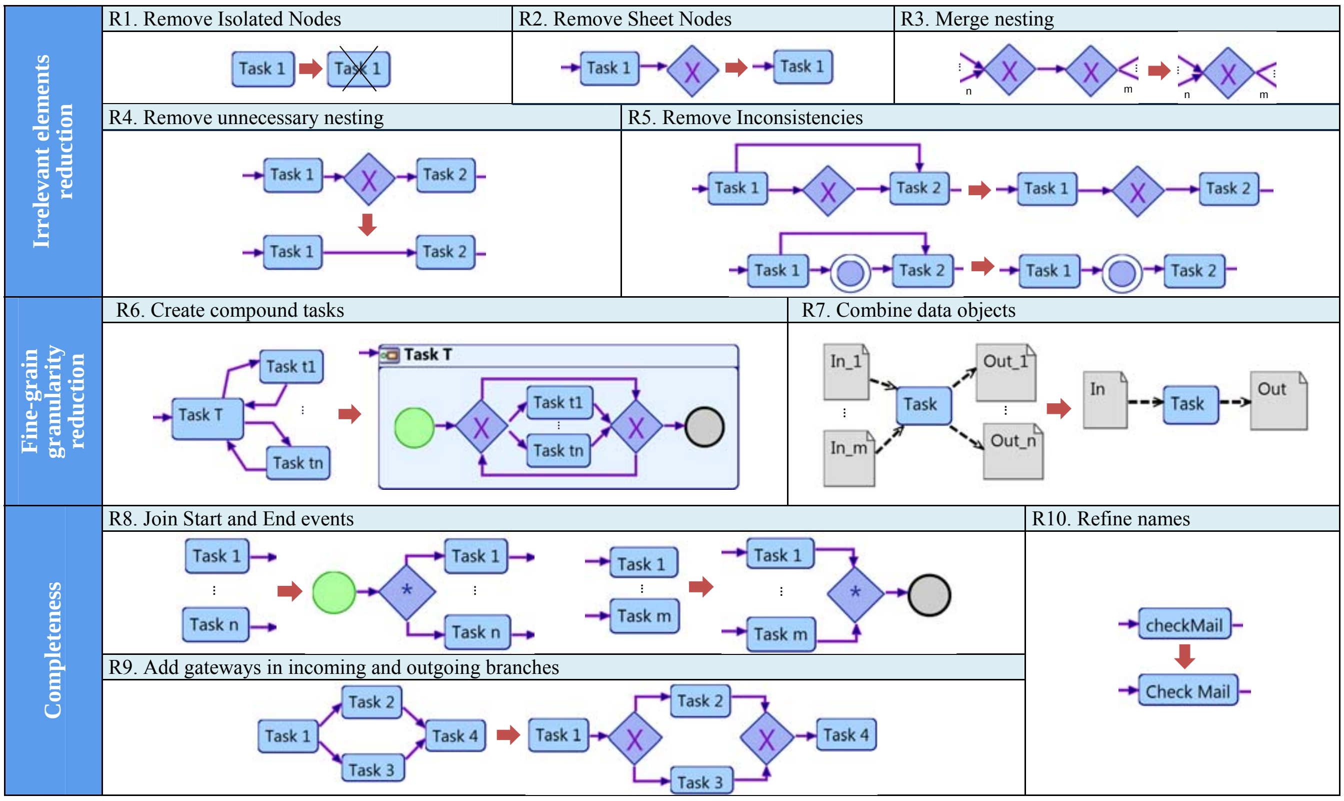
3. IBUPROFEN

IBUPROFEN (Improvement and BUsiness Process Refactoring OF Embedded Noise) is a framework with which to refactor business process models particularly retrieved by reverse engineering. IBUPROFEN is specially designed for business process models represented according to the BPMN (Business Process Modeling Notation) [11]. IBUPROFEN allows applying different refactoring operators taking into account the assessment of various measures related to the modifiability and understandability of business process models [4].

IBUPROFEN is supported by a tool that has being implemented as an Eclipse^{TM} plug-in [12]. The supporting tool can therefore be used in combination with other Eclipse™ plug-ins aimed at obtaining business process models from the source code of existing information systems.

IBUPROFEN provides a set of ten refactoring operators (see Table 1) grouped into three categories: irrelevant elements reduction, fine-grain granularity reduction and completeness. The next sub-sections explain in detail each refactoring operator.

Table 1. Refactoring operators



3.1 Irrelevant Elements reduction

This category groups the refactoring operators responsible for removing irrelevant elements found in business process models like isolated tasks, sheet tasks and inconsistencies. Moreover, nested gateways can origin an increase in the complexity of business process models, thus these are replaced by equivalent, light-weight structures. The refactoring operators are the following:

- **R1. Remove Isolated Nodes:** This refactoring operator removes nodes (i.e., tasks, gateways or events) in the business process model that are not connected with any other node in the business process model.
- **R2. Remove Sheet Nodes:** This refactoring operator removes elements in the business process model that are considered sheet nodes. These nodes can be gateways or intermediate events that have no successor nodes.
- **R3. Merge nesting:** This refactoring operator merges consecutive gateways of the same type when the first gateway has only one output and the second has only one input.
- **R4. Remove unnecessary nesting:** This refactoring operator removes gateways that connected only two nodes, i.e. with one input and one output. This gateway is removed and a direct sequence flow is created between these nodes.
- **R5. Remove Inconsistencies:** This refactoring operator removes sequence flows in the business process model that are considered as inconsistent. When two tasks are connected through a cut node, as an intermediate event or a gateway, and through a direct sequence flow this sequence flow are removed.

3.2 Fine-grain granularity reduction

The different granularity of business tasks and callable units in existing information systems constitutes another important challenge [13]. According to the approach proposed by [14], each callable unit in an information systems is considered as a candidate business task. However, existing systems typically contain thousands of callable units, some of which are large ones supporting the main business functionalities of the system, while many are very small and do not directly support any business activity. In other situations a set of small callable units together supports a business activity. As a consequence, this category provides two refactoring operators to deal with large sets of fine-grain business tasks and data objects:

- **R6. Create compound tasks:** This refactoring operator transforms each task in a compound task when the task T has several subsequent tasks which are in turn connected with a round-trip sequence flow to the task T. This scenario is due to each callable unit is transformed as a task during the reverse engineering stage when a certain callable unit can invoke another callable unit returning a value to the first one. In this case, the refactoring operator creates a compound task with a start and end event connected with each subsequent task through the respective split and join exclusive gateways.
- **R7. Combine data objects:** This refactoring operator combines data objects that are input and/or output of a task. The combination is possible when those data objects are exclusively used (written or read) for that task. The combination is done when the number of data objects is above a threshold. In order to mitigate the collateral semantic loss, all the names of the grouped data objects are saved in the documentation attribute provided by the BPMN standard.

3.3 Completeness

Any reverse engineering technique implies an increase of the abstraction degree, and therefore a semantic loss. For this reason, this category of operators is provided to deal with semantic loss by means of the incorporation of additional elements. The refactoring operators are the following:

- **R8. Refine names:** This operator implements a heuristic to improve task labels of business tasks that were obtained almost directly from methods or functions of legacy source code through reverse engineering. These kinds of labels consist of the concatenation of various capitalized words, in accordance with naming conventions present in most programming approaches. In an effort to have more understandable names, this refactoring operation split these labels into ones with various words.
- **R9. Join Start and End events:** This refactoring operator joins the start and end event with the starting and ending tasks, respectively. These events are created whether such events were not created by reverse engineering. When there are several starting tasks the refactoring operator adds a split complex gateway between the start event and starting tasks. Similarly, if there are several ending tasks, the refactoring operator adds a join complex gateway between ending tasks and the end event [15].
- **R10. Add gateways in incoming and outgoing branches:** As a result of the use of reverse engineering to retrieve business process models, it is possible to obtain models that do not follow some of the good modeling practices according the BPMN standard with regard to the usage of gateways. In this case, this operator adds a join and split exclusive gateways when a certain task respectively has several precursor or subsequent tasks.

4. CASE STUDY

This section provides a case study with a real-life information system. The case study has been conducted by following the formal protocol developed by *Runeson et al*. [16] for conducting and reporting case studies in the software engineering field. The following sections show the stages proposed in the formal protocol: the background, case study design, case selection procedure, execution procedure and data collection, analysis and interpretation, and finally, threats to the validity.

4.1 Background

The *object* of this case study is IBUPROFEN and the *purpose* of this case study is to evaluate how the execution order of the different refactoring operators influences the quality degree of the target business process models in terms of understandability and modifiability. Taking into account the object and purpose of the study the main research question is proposed as follows:

RQ1: *Does the order of the application of refactoring operators' categories affect the understandability and modifiability of business process models?*

4.2 Case Study Design

The case study follows the *embedded case study* design according to the classification proposed by *Yin* [17], whereby the case study consists of a single case (i.e., it focuses on a single information systems) but considers several analysis units as *independent variable* within the case. The analysis units in this study are the different business processes models retrieved from an information system. Therefore, the study consists of applying all the refactoring operators in different orders and obtaining business processes that are in turn analyzed to answer RQ1. For this purpose, some measures as *dependent variables* are established to quantitatively answer RQ1 in terms of understandability and modifiability of business process models [3].

- **Size**: This measure is the number of nodes in a business process model (i.e., business tasks, gateways, data objects and events). This measure affects negatively to the understandability, i.e. a higher size difficult the understandability of a certain business process model.
- **Connectivity**: This measure is the ratio between the total number of arcs in a business process model (i.e., sequence flows and associations) and the total number of nodes. This measure negatively affects the understandability and modifiability. This signifies that lower connectivity values imply business process models more understandable and modifiable due to a lower intricacy.
- **Separability**: This measure represents the ratio between the number of cut-vertices in a business process model (i.e. nodes that serve as bridges between otherwise strongly-connected components) and the total number of nodes in the business process model. This measure affects negatively to the modifiability since a higher separability implies hard and error-prone modifications of business process models.

4.3 Case Selection Procedure

To select the case under study a set of selection criteria were formulated in order to rigorously select the source system: (1) the system should be a real-life information system currently in production; (2) the size of the system should be greater to 20 KLOC (thousands of lines of source code) to ensure that the system supports a great number of business processes; (3) the system should be written in Java language to be able to use the MARBLE tool. MARBLE is a business process archeology tool. This tool was selected because is released as an Eclipse plug-in and it therefore can be easily integrated with the IBUPROFEN tool.

After analyzing a dozen of information systems of some partner companies according to these criteria, the selected case was AELG-Members, which is a real-life system used by *the Association of Writers in Galician Language* in Spain. AELG-Members is an author management system to support the registration, fee payments, among other of the mentioned organization. From a technological viewpoint, AELG-Members is a standalone Java application whose architecture follows the traditional structure in three layers [18]: (1) the *domain* layer supporting all the business entities and controllers; (2) the *presentation* layer dealing with the user interfaces; and (3) the *persistency* layer handling data access. The total size of the legacy system is 23.3 KLOC.

4.4 Execution Procedure and Data Collection

The procedure to be performed to execute the case study consists of a set of steps which are numbered below:

1. Business process models are mined from the source code using MARBLE [19].
2. A set of six different execution orders of refactoring operators are selected to be executed in the IBUPROFEN tool. These six different orders are defined with all the possible combinations with the three refactoring operator categories (i.e., irrelevant, granularity and completeness) (see Table 2).
3. The set of orders are executed using the IBUPROFEN tool and the values of the mentioned measures are collected for each business process model. IBUPROFEN is executed in a computer with a 2.66 GHz dual processor and 4.0 GB RAM.
4. After the whole execution, the collected information (see Table 3, Table 4 and Table 5) is statistically analyzed to answer the main research question. Additionally, a univariant general linear model test was made to demonstrate the effect of the execution order.

Table 2. Set of orders

Order Id	Refactoring Operators		
1	Irrelevant	Granularity	Completeness
2	Irrelevant	Completeness	Granularity
3	Completeness	Irrelevant	Granularity
4	Completeness	Granularity	Irrelevant
5	Granularity	Completeness	Irrelevant
6	Granularity	Irrelevant	Completeness

Table 3. Data collected for the size measure

BP	Order ID						
	Original	1	2	3	4	5	6
1	6	3	6	9	9	6	3
2	28	17	25	37	37	29	17
3	1	0	0	3	3	3	0
4	49	29	29	41	39	41	27
5	15	0	0	12	12	12	0
6	1	0	0	3	3	3	0
7	15	5	5	16	16	16	5
8	93	31	31	61	61	61	31
9	46	13	13	29	29	29	13
10	144	272	277	282	282	277	272
11	3	3	1	1	0	3	0
12	123	21	76	95	95	27	16
13	137	56	82	109	109	48	31
Mean	50.85	34.62	41.92	53.69	53.46	42.69	31.92
S. Dev.	54.37	73.19	75.85	76.95	77.04	72.81	73.11

Table 4. Data collected for the connectivity measure

BP	Order ID						
	Original	1	2	3	4	5	6
1	1.000	1.000	1.167	1.000	1.000	0.833	1.000
2	0.643	1.235	1.480	1.622	1.622	1.517	1.235
3	0.000	0.000	0.000	0.667	0.667	0.667	0.000
4	1.041	2.034	1.828	1.829	1.923	1.976	2.037
5	0.000	0.000	0.000	1.500	1.500	1.500	0.000
6	0.000	0.000	0.000	0.667	0.667	0.667	0.000
7	0.200	0.800	0.800	1.500	1.500	1.500	0.800
8	0.237	0.968	0.968	1.426	1.426	1.426	0.968
9	0.630	1.385	1.385	1.724	1.724	1.724	1.385
10	5.194	4.746	4.697	4.645	4.645	4.693	4.746
11	0.667	0.667	0.000	0.000	0.000	0.667	0.000
12	1.659	1.476	1.868	1.768	1.768	1.556	1.375
13	1.562	1.714	1.805	1.963	1.963	1.792	1.774
Mean	0.987	1.233	1.231	1.562	1.570	1.578	1.178
S. Dev.	1.383	1.241	1.275	1.088	1.090	1.041	1.281

Table 5. Data collected for the separability measure

BP	Order ID						
	Original	1	2	3	4	5	6
1	4	2	2	5	5	5	2
2	22	13	13	25	25	25	13
3	1	0	0	3	3	3	0

BP	Order ID						
	Original	1	2	3	4	5	6
4	31	17	17	29	27	29	15
5	15	0	0	12	12	12	0
6	1	0	0	3	3	3	0
7	14	5	5	16	16	16	5
8	93	31	31	61	61	61	31
9	46	13	13	29	29	29	13
10	30	11	11	21	21	21	11
11	2	3	1	1	0	3	0
12	36	10	7	20	20	21	10
13	60	17	17	32	32	36	19
Mean	27.31	9.38	9.00	19.77	19.54	20.31	9.15
S. Dev.	27.01	9.09	9.24	16.47	16.49	16.58	9.33

4.5 Analysis and interpretation

After the execution of the case study, the measure values for size, connectivity and separability were collected for 13 business process models. The data analysis performed is as follows.

The mean of size was between 31.92 (order 6) and 53.69 (order 3). These values entail between a decrease of 37.22% and an increase of 5.58% with regard to the original size. Thus, in general terms, the understandability of business process models is improved in all orders except order 3 and 4, being the order 6 the best order taking account the size variable.

The mean of connectivity was between 1.18 (order 6) and 1.58 (order 5). These values entail an increment between 16.10% and 37.66% with regard to the original connectivity value. Therefore, in term of connectivity all the orders do not improve understandability or modifiability. This is due to the additive operators (e.g., R8 and R9) which incorporate additional elements to the business process model under refactoring. Anyway, the best choice is order 6, since provides the minimum difference of connectivity.

The mean of separability was between 9.0 (order 2) and 20.31 (order 5). These values entail a decrease between 67.04% and 25.63% with regard to the original size. Consequently, in general terms, the modifiability of business process models is improved in every order, being the order 5 the best order for the separability.

Figure 1 shows a boxplot of all the measures taken for the six orders. Since all the measures values are normalized, Figure 1 shows a line in zero to represent improved results (under the line) and exacerbate results (above the line). The size and separability values are under the line while the connectivity values are close to cero since the order is not relevant.

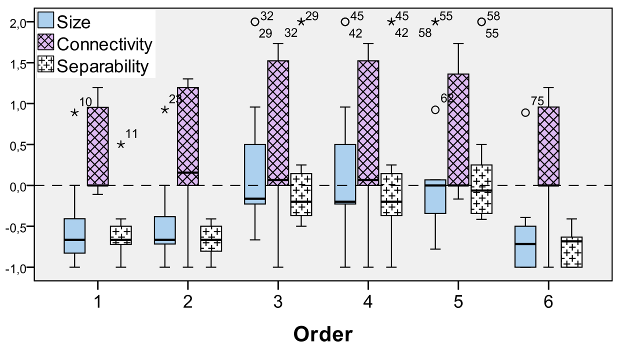


Figure 1. Boxplot of size, separability and connectivity

To answer RQ1, a univariant linear model test was applied to determine the influence of the execution order. In order to carry out this test, all the measure values had to be necessarily normalized using the equation $z = x / y - 1$, being z the normalized value, x the value of the variable to normalize and y the original value of this variable. The hypotheses of the test were:

H_0: The execution order of refactoring operator categories has no effect in the quality degree of business process model.

H_1: ¬H0: The execution order of refactoring operator categories has effect in the quality degree of business process model.

After the application of the test, in the case of the size and separability, the value of the significance test was less than 0.005, (0.001 and 0, respectively). This means that it is not possible to reject the null hypothesis and therefore the alternative one is accepted. However, in case of the connectivity the value of the significance test was 0.906 so the null hypothesis is accepted. This signifies that the execution order is relevant in the final quality degree of business process models in terms of size and separability but not in terms of connectivity. Once the research question was answered, the best execution was the sixth order taking into account the three variables as a whole, i.e., reducing the granularity, then removing irrelevant elements and lastly solving the completeness problems.

4.6 Threats to the validity

This section presents the threats to the validity of this case study and possible actions to mitigate them. There are mainly three types of validity: internal, construct and external.

As far as the *internal* validity is concerned, a sample of 13 business process models was retrieved from a sole information system, and it is thus possible to obtain statistically representative results. Nevertheless, the study may be replicated by using more information systems, to attain a larger sample of business process models.

In addition, there are two decisive threats. The first one is related to the way in which business processes models were retrieved by reverse engineering. MARBLE, the supporting tool used to obtain the business process models, could be a factor that affects the initial sample of business process models. Secondly, the set of refactoring operators included in IBUPROFEN is a threat to the generalization of the results. The replication of the study using different refactoring operators and different refactoring techniques may be a mean for mitigating these threats.

Moreover, with respect to the *construct* validity, the selected measures (size, connectivity and separability) were suitable for assessing the quality of business process models in terms of their understandability and modifiability. However, there are other measures in literature that directly affects the understandability and modifiability such as density, complexity, average of gateway degrees, among other [20-24]. For this reason, the effect of these additional measures should be evaluated in the future.

Other threat to the construct validity is that the execution order analysis was made by only considering all the possible combination between the three categories of refactoring operators (irrelevant, granularity, completeness). However, the particular order of each refactoring operator has not been assessed in the case study. Since all the possible combinations for 10 refactoring operators will lead to check 10! combinations (i.e., 3,628,800). In order to address this study, some pre- and post-conditions will defined in the future for pruning non-promising combinations. It allows us to evaluate all the relevant combinations and assess the execution order by considering the refactoring operations instead of the category.

Finally, *external* validity is concerned with the generalization of the results. This study considers the whole population to be business process models retrieved by reverse engineering from legacy information systems. The results obtained can be strictly generalized to this population with the particularity that all the information systems under study are based on Java platform. This restriction is related to the supporting tools used in this study (MARBLE and IBUPROFEN). This threat may be mitigated by replicating the study using systems implemented in different platforms which are additionally analyzed with different refactoring tools and methods.

5. CONCLUSIONS

Appropriate description of business processes through standard notations has become one of the most important assets for organizations but these can contain faults that affect their quality. For this reason, this paper presents IBUPROFEN, a refactoring framework with which to improve the quality of business process models, particularly for the case of model mined by reverse engineering. IBUPROFEN applies different refactoring operators taking into account the assessment of various measures related to the modifiability and understandability of business process models. To support the technique, an Eclipse plug-in has been created with the same name to implement the entire refactoring operators. All the refactoring operators are group in three groups according with their behavior: irrelevant elements reduction, fine-grain granularity reduction and completeness.

Unfortunately, refactoring techniques in the literature do not attempt to define best-order execution of the refactoring operators to ensure the best gain of understandability and modifiability. To address the aforementioned challenge, the paper presents a case study to assess the effect of different execution orders of refactoring operators in the final quality degree of business process models. For this purpose, size, connectivity and separability measures were considered to evaluate the understandability and modifiability of business process models. The statistical hypothesis test results demonstrated that the execution order affects understandability and modifiability of business process models. The main implication is that the best execution order to improve understandability and modifiability is the sixth order, i.e., reducing the granularity, then removing irrelevant elements and lastly solving the completeness problems.

According to the mentioned threats to the validity, the work-in-progress of this proposal is consisting of the replication of this case study with additional information systems in order to figure out the best execution order of refactoring operator instead of their categories.

6. ACKNOWLEDGMENTS

This work was supported by the FPU Spanish Program and the R&D projects GEODAS (TIN2012-39493-C03-01), ALTAMIRA (PII2I09-0106-2463), PEGASO /MAGO (TIN2009-13718-C02-01), MOTERO (PEII11-0366-9449) and ARMONIAS (PII2I09-0223-7948).

7. REFERENCES

[1] Weske, M., Business Process Management: Concepts, Languages, Architectures. 2007, Leipzig, Germany.

[2] van der Aalst, W.M.P., A.H.M.T. Hofstede, and M. Weske, Business process management: a survey, in Business process management. 2003, Eindhoven, The Netherlands. p. 1-12.

[3] Fernández-Ropero, M., et al., Quality-Driven Business Process Refactoring, in International Conference on Business Information Systems (ICBIS 2012). 2012: Paris, France. p. 960-966.

[4] Fernández-Ropero, M., R. Pérez-Castillo, and M. Piattini, Refactoring Business Process Models: A Systematic Review, in 7th International Conference on Evaluation of Novel Approaches to Software Engineering: Wrocław, Poland. p. 140-145.

[5] Weber, B., et al., Survey paper: Refactoring large process model repositories. Comput. Ind., 2011. 62(5): p. 467-486.

[6] Dijkman, R., M.L. Rosa, and H.A. Reijers, Managing large collections of business process models—Current techniques and challenges. Computers in Industry, 2012. 63(2): p. 91.

[7] Dijkman, R., et al., Identifying refactoring opportunities in process model repositories. Information and Software Technology, 2011.

[8] La Rosa, M., et al., Managing process model complexity via abstract syntax modifications. Industrial Informatics, IEEE Transactions on, 2011. 7(4): p. 614-629.

[9] Leopold, H., S. Smirnov, and J. Mendling, Refactoring of process model activity labels, in Proceedings of the Natural language processing and information systems, UK. p. 268-276.

[10] Gambini, M., et al., Automated error correction of business process models. Business Process Management, 2011: p. 148-165.

[11] OMG. Business Process Modeling Notation Specification 2.0. 2011; Available from: http://www.omg.org/spec/BPMN/2.0/PDF/.

[12] Alarcos Research Group. IBUPROFEN. 2012; Available from: http://marketplace.eclipse.org/node/423052.

[13] Pérez-Castillo, R., et al., Generating Event Logs from Non-Process-Aware Systems Enabling Business Process Mining. Enterprise Information System Journal, 2011. 5(3): p. 301–335.

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

[15] Mendling, J., H.A. Reijers, and W.M.P. van der Aalst, Seven process modeling guidelines (7PMG). Information and Software Technology, 2010. 52(2): p. 127-136.

[16] Runeson, P. and M. Höst, Guidelines for conducting and reporting case study research in software engineering. Empirical Softw. Eng., 2009. 14(2): p. 131-164.

[17] Yin, R.K., Case study research. Design and methods. 3rd ed. 2003, London: Sage.

[18] Eckerson, W., Three Tier Client/Server Architecture: Achieving Scalability, Performance and Efficiency in Client Server Applications. Open Information Systems, 1995. 10(1): p. 3.

[19] Pérez-Castillo, R., et al., MARBLE. A Business Process Archeology Tool, in 27th IEEE International Conference on Software Maintenance (ICSM 2011). 2011: Williamsburg, VI. p. 578 - 581

[20] Cardoso, J. Process control-flow complexity metric: An empirical validation. 2006: IEEE.

[21] Rolon, E., et al. Prediction models for BPMN usability and maintainability. 2009: IEEE.

[22] Sánchez-González, L., et al., Quality assessment of business process models based on thresholds. On the Move to Meaningful Internet Systems: OTM 2010, 2010: p. 78-95.

[23] Reijers, H.A. and J. Mendling, A study into the factors that influence the understandability of business process models. Systems, Man and Cybernetics, IEEE Transactions on, 2011(99): p. 1-14.

[24] Mendling, J. and M. Strembeck. Influence factors of understanding business process models. 2008: Springer.