Information and Software Technology 53 (2011) 937–948

Contents lists available at ScienceDirect

Information and Software Technology

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

Identifying refactoring opportunities in process model repositories

Remco Dijkman [a,*], Beat Gfeller [b], Jochen Küster [b], Hagen Völzer [b]

[a] Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, The Netherlands
[b] IBM Research – Zurich, Säumerstrasse 4, 8803 Rüschlikon, Switzerland

ARTICLE INFO

Article history:
Available online 9 April 2011

Keywords:
Business Process Model
Refactoring
Repository

ABSTRACT

Context: In order to ensure high quality of a process model repository, refactoring operations can be applied to correct anti-patterns, such as overlap of process models, inconsistent labeling of activities and overly complex models. However, if a process model collection is created and maintained by different people over a longer period of time, manual detection of such refactoring opportunities becomes difficult, simply due to the number of processes in the repository. Consequently, there is a need for techniques to detect refactoring opportunities automatically.
Objective: This paper proposes a technique for automatically detecting refactoring opportunities.
Method: We developed the technique based on metrics that can be used to measure the consistency of activity labels as well as the extent to which processes overlap and the type of overlap that they have. We evaluated it, by applying it to two large process model repositories.
Results: The evaluation shows that the technique can be used to pinpoint the approximate location of three types of refactoring opportunities with high precision and recall and of one type of refactoring opportunity with high recall, but low precision.
Conclusion: We conclude that the technique presented in this paper can be used in practice to automatically detect a number of anti-patterns that can be corrected by refactoring.

1. Introduction

Today, many organizations maintain repositories that contain hundreds of business process models. For example, the SAP reference model [5], the reference model for Dutch local government [9] and IBM's insurance architecture (IAA) [15] all contain more than 250 process models.

To promote a joint understanding of the repository and, therewith, re-use and maintainability, the processes in the repository should be modeled as uniformly as possible. In particular, they should use the same terms to describe model elements that have the same meaning and different terms to describe model elements that have a different meaning. To promote maintainability, overlap between processes should be avoided; process parts that appear in multiple business process models should be put into common subprocesses, such that a change to these parts only has to be made in one place. To promote uniformity, the models in the repository should be described at the same level of detail. Refactoring operations can be applied to the process models in a repository to fix problems related to terminology, overlap and level of detail and, therewith, to promote understandability, re-use and maintainability of the models [33]. By *refactoring*, we mean an operation that changes one or more business process models in a repository, without changing their execution semantics, but improving their understandability, maintainability or reusability.

Refactoring may not only apply to the process models, but also to the real-world processes themselves, saving costs by eliminating redundancies in their implementations.

This paper presents a technique to automatically detect opportunities for applying refactoring operations to fix problems related to terminology, overlap and level of detail. For some problems such as activity naming inconsistencies, simple process matching techniques [1,7] suffice to detect a refactoring opportunity. However, some refactoring opportunities concern similar process parts, i.e., coherent groups of several related nodes. To find similar process parts, our approach combines process matching techniques with the Refined Process Structure Tree [32], which can be used to group related nodes in a computationally efficient manner.

Identifying similar parts of different processes also has value on its own, without necessarily aiming at refactoring. For example, it can refine the search for similar processes, which is used, e.g., to find re-usable assets that are associated with a similar process, such as a software application or an expert in a role associated with that process. There may be no pair of processes that are similar overall, but there may be processes that have similar parts. Techniques exist to search for similar processes [6]. However, searching

* Corresponding author. Tel.: +31 40 2474370; fax: +31 40 2432612.
E-mail addresses: r.m.dijkman@tue.nl (R. Dijkman), bgf@zurich.ibm.com (B. Gfeller), jku@zurich.ibm.com (J. Küster), hvo@zurich.ibm.com (H. Völzer).

0950-5849/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2011.04.001

similar process parts is in general computationally harder, because it requires that a process is split up into parts and the number of potentially relevant process parts can be exponential in the number of its elements.

This paper has two contributions. Firstly, we show how existing techniques can be combined to efficiently detect many refactoring opportunities in practice. In addition, we present, based on that technique, different metrics to determine the type of problem that the similarity represents, and therewith the type of the refactoring that can be applied. Secondly, we evaluate our technique, by applying it to two real-world reference process model collections. Our evaluation shows that similar process parts occur frequently in those collections and that our similarity metrics can indeed be used for identifying refactoring opportunities.

Our technique focusses only on one aspect of a business process model, viz. on the activities and the groups they form. This restricts our attention to certain important refactorings but we can obtain satisfactory results with a relatively simple technique. Simplicity is not only helpful to transfer a technique to practical application, but focussing on that one aspect also makes it easier to apply the technique across different modeling languages and styles.

The remainder of this paper is organized as follows. Section 2 defines the refactoring opportunities that we aim to detect. We describe the Refined Process Structure Tree (which originates from [32,23]) in Section 3 as a basis for determining similarity of process parts. In Section 4, we define different similarity metrics (Section 4.1, which builds on previous work, and Section 4.2, which is new), present a technique for computing similarity of process parts (Section 4.3) and explain its relationship to refactoring opportunities (Section 4.4). Section 5 reports on the results obtained when applying the technique to a repository of process models. Section 6 presents related work, before Section 7 concludes the paper.

2. Refactoring opportunities in model collections

This section defines opportunities for four refactoring operations which improve the quality of a collection of business process models. The focus of this paper is on automatically identifying such refactoring opportunities in a collection. Consequently, we derived the refactoring opportunities by first studying refactoring operations that have been defined in previous work [33,11,17]; second, determining the situations in which they can be applied; and, third, selecting the situations that can be detected automatically as *refactoring opportunities*.

Fig. 1 shows the four refactoring opportunities that we have identified in this manner. For each opportunity, it also shows the refactoring operation that can be applied to it in order to create a process model collection that is easier to understand, maintain and re-use.

The first refactoring opportunity is the situation in which there are activities ('Check claim' and 'Verify claim' in the figure) that are considered to be the same (indicated by the dashed line), but that have different labels. In this situation, a renaming operation should be applied to give the activities uniform labels. The decision as to whether or not the two activities are the same should be made by a human observer based on, for example, brief descriptions of the activities or knowledge about what the activities represent in practice.

The second refactoring opportunity is the situation in which there are process fragments (defined precisely in the next section) that are, as a whole, similar, because the activities and control-flow relations that they are composed of are similar. In this situation the fragment from one of the process models should substitute the other fragment and should be extracted into a common subprocess. Figs. 2a and c show two similar process fragments F_{a2} and F_{c2} which can be extracted into such a common subprocess.

The third refactoring opportunity is the situation in which there are process fragments that are similar, but are composed of both similar activities and activities that appear in one but not in the other fragment. In this case a common subprocess should replace the fragments. This subprocess should be composed of both the similar and the non-similar activities and have the option to skip the non-similar activities, because these activities can be performed in one process but not in the other. Optionally, the ability

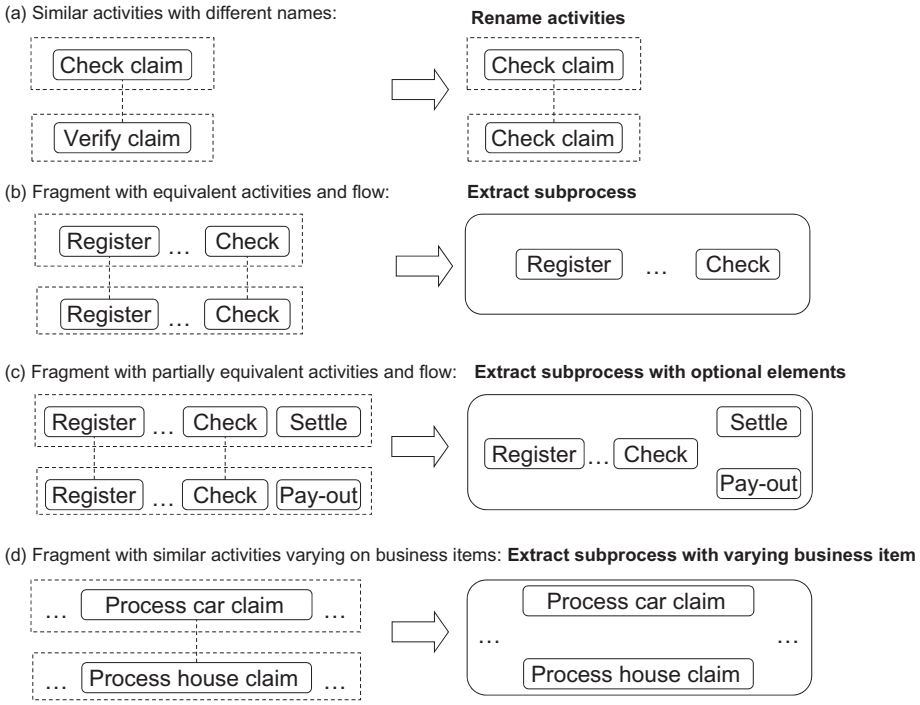

Fig. 1. Refactoring opportunities and operations.