Information and Software Technology 80 (2016) 20–35

Contents lists available at ScienceDirect

Information and Software Technology

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

Evolving process views

Rik Eshuis[a,*], Alex Norta[b], Raoul Roulaux[a]

[a] Eindhoven University of Technology, The Netherlands
[b] Department of Informatics, Tallinn University of Technology, Estonia

ARTICLE INFO

Article history:
Received 23 June 2015
Revised 11 August 2016
Accepted 11 August 2016
Available online 12 August 2016

Keywords:
Inter-organizational
Business process management
Process evolution
Process visibility
Smart contract
E-governance
Decentralized autonomous organization

ABSTRACT

Context: Process views support the paradigm of Business Process Outsourcing, in which providers perform business processes on behalf of their clients. A public process view shields secret or irrelevant details from a private, internal business process, thus allowing a provider to reveal only relevant, non-confidential parts of its business process to its clients. Providers can change their internal business processes that may result in inconsistencies with the corresponding process views.

Objective: This paper aims to develop an approach for propagating changes from an internal, private process to its public process view, such that the internal process and its process view remain consistent.

Method: We develop the approach in a formal way. Definitions of process models and process views are based on BPEL, the standard language for realizing process models using state-of-the-art service-oriented technology. We validate the feasibility of the approach by showing how it can be supported by a conceptual system architecture.

Results: The approach relies on two key results. First, a formal characterization of the set of private changes to an internal process, i.e., changes that do not need to be propagated to the process view. Second, a characterization of the non-private changes that can safely be propagated from an internal process to its process view such that they remain consistent. Other non-private changes result in an internal process and a process view that are not consistent. The approach is supported by a system architecture for process-based business collaboration.

Conclusion: The approach supports providers in deciding if and how changes to a private, internal process propagate to its public process view such that the process view and the internal process remain consistent. The approach allows clients to monitor a public process view such that they can safely track changes made to a private, internal process.

1. Introduction

Business Process Outsourcing is a paradigm in which a provider performs, or coordinates a process on behalf of its client, the consumer. The outsourced process is note a blackbox for the consumer, as the consumer needs status information to coordinate its other processes. Some parts of the performed process in the provider domain may be confidential, or irrelevant for the consumer. To solve this problem, the notion of process views has been proposed [1–3]. A process view is an abstraction of an internal process, thus allowing consumers to monitor the progress of process instances, while at the same time shielding private or irrelevant parts of the provider process from the consumer. Process views are also used to visualize different end-user perspectives on internal processes [4].

Process views can be defined in two different ways: as virtual or materialized entities. Virtual process views are projections of internal processes: they do not exist as entities independent from internal processes [1,5]. However, in the context of outsourcing, it is natural to consider *materialized* process views [6,7], which exist independent from the internal processes they abstract from. Materialized process views correspond to smart contracts [8,9] that specify the work to be performed on behalf of the consumer [11]. A smart contract is a legally enforceable agreement in which two or more parties commit to certain obligations in return for certain rights [12]. Contracts are instruments for organizing business collaborations. Smart contracts aim at using information technologies to significantly improve the efficiency and effectiveness of paper contracts, allowing companies to support newly emerging business paradigms such as for cyber-physical systems [10], while still being legally protected. Moreover, using materialized process views helps to establish outsourcing relationships by supporting matchmaking between offered and requested process views [13,14]. Notably, materialized process views are an integral part of the

* Corresponding author.
E-mail addresses: h.eshuis@tue.nl (R. Eshuis), alex.norta.phd@ieee.org (A. Norta).

http://dx.doi.org/10.1016/j.infsof.2016.08.004
0950-5849/

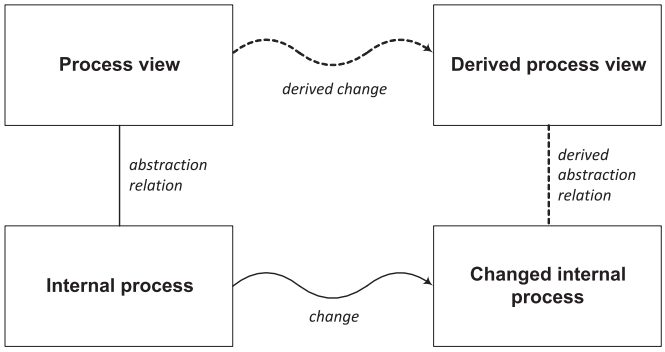

Fig. 1. Overview of the approach: how to propagate a change of an internal process to a change of its materialized process view?

eSourcing-framework [13,15] for harmonizing on an external layer the intra-organizational business processes of a service consuming and one or many service providing organizations into a B2B supply-chain collaboration. Important elements of eSourcing are the support of different visibility layers of corporate process views for the collaborating counterpart and flexible mechanisms for service monitoring and information exchange.

A key feature of business processes in modern organizations is that they frequently change [16]. Process-oriented information systems are increasingly capable of supporting such changes [17]. If an internal process changes, the process view needs to be aligned with the change. If the process view is materialized, this means the view is changed based on the change in the internal process. Such a procedure is similar to amending a contract between two parties by explicitly mentioning the change applied to the old contract, allowing the client to safely track the change. Current process view approaches [2–7,13,18–21] do not support the propagation of changes made to internal processes to changes in materialized process views.

To fill this gap, this paper presents a formal approach that defines how changes to an internal process propagate to changes to a materialized process view such that both remain consistent (see Fig. 1). Input to the approach is a change to an internal process that has a consistent process view, i.e., there exists an abstraction relation between them. The approach defines the resulting change to the process view and the resulting change to the abstraction relation such that a changed internal process view and the changed view are consistent again. Thus, the changed process view is an abstraction of the changed internal process.

The approach distinguishes between *private* and *public* (non-private) changes. Private changes to an internal process do not affect the view, so these changes result in empty changes for the view. We formally characterize which changes are private and which are public. We show that for some of the public changes, the changed view is not an abstraction of the changed internal process. Such changes should be blocked or modified. Finally, we formally characterize the subset of public changes to an internal process that can be safely propagated to its process view. We validate the feasibility of the approach by showing how it can be supported by a conceptual system architecture.

We focus in this paper on structured processes [22], since they are close to BPEL [23], which is the standard language for expressing process views using state-of-the-art service-oriented technology. Moreover, structured processes allow for a concise and simple definition of the change operations as well as the abstraction relation between an internal process and its process view. Several approaches have been defined to convert an unstructured into a structured process while preserving its behavior [24,25].

This paper is organized as follows. Section 2 discusses related work. Section 3 presents a motivating example that illustrates the problem of propagating changes from internal processes to process views. Section 4 formally defines structured processes as process trees. We also define several change operations on process trees in a concise manner. Next, Section 5 formally defines process views. Section 6 formally characterizes private changes to an internal process that do not affect a process view, which abstracts the internal process. Section 7 characterizes under which conditions and how public (non-private) changes to an internal process can be propagated to a view. The other applied public changes result in an inconsistent view and therefore should be blocked. Section 8 validates the feasibility of the approach and Section 9 discusses the findings of this paper. Finally, Section 10 presents conclusions and future work.

2. Related work

Process views have received increased research attention in recent years in the context of inter-organizational business process management [1–3,6,7,13,14,18–21,26–29], where a provider performs a business process on behalf of a consumer. Most of these papers deal with the design-time aspects of process views [2,3,7,13,18–21,26–29]. Most process view design approaches focus on deriving a public process view from a private, internal process [2–4,7,13,14,20,21,26,28,29] while others study the derivation of internal processes from public views [2,18,19]. Other papers [1,6] focus on how to support a process view at run-time and do not address how to construct a process view. All these papers consider static process views, so the constructed process views, once deployed, do not change. Process views have also been applied to offer personalized, role-specific views on a common process [4,5,86]. Smirnov et al. [30] provide a survey of different static abstraction approaches. All these papers study the relation between an internal process and a process view, but not consider change propagation from internal processes to process views, which is studied in this paper.

From a conceptual point of view, process views and their underlying internal process models are expressed in separate models. From an implementation point of view, process views can be implemented in separate models [7,29,31]. We call such process views *materialized*. Alternatively, process views can be implemented as projections of internal processes [1,5]. In that case, process views are virtual, as they are defined by abstraction relations defined on top of internal process models. In this paper, we focus on materialized process views, as we motivated in Section 1.

Other related approaches that deal with changes in the context of process views [5,32,33] all consider virtual process views. For a virtual process view, the abstraction relation defines the process view. Therefore, changing the abstraction relation results in a new virtual process view. However, we consider materialized process views. In that case, a private change to an internal process needs to be propagated to a corresponding public change to its materialized process view. This separation between private and corresponding public changes allows a consumer to safely track in the public process view public changes that abstract from private changes made to confidential parts of internal processes. A complication that arises for materialized process views is that an internal process and its process view may no longer be consistent after applying changes. This motivates a characterization of internal changes that cannot be propagated to process views, as we did in this paper. This issue is not applicable to virtual process views [5,32,33].

Weidlich et al. [34] study how to propagate changes between aligned process models. Given a change in a process model, they focus on identifying the relevant part of the aligned process model