# Goal Oriented Variability Modeling in Service-Based Business Processes

Karthikeyan Ponnalagu[1,3], Nanjangud C. Narendra[2], Aditya Ghose[3], Neeraj Chiktey[1,4], and Srikanth Tamilselvam[1]

[1] IBM Research India, Bangalore, India
{pkarthik,srikanth.tamilselvam}@in.ibm.com
[2] IBM India Software Lab, Bangalore, India
narendra@in.ibm.com
[3] University of Wollongong, Australia
aditya.ghose@gmail.com
[4] International Institute of Information Technology, Hyderabad, India
chikteyneeraj@yahoo.co.in

**Abstract.** In any organization, business processes are designed to adhere to specified business goals. On many occasions, however, in order to accommodate differing usage contexts, multiple variants of the same business process may need to be designed, all of which should adhere to the same goal. For business processes modeled as compositions of services, automated generation of such *goal preserving* process variants is a challenge. To that end, we present our approach for generating all goal preserving variants of a service-based business process. Our approach leverages our earlier works on semantic annotations of business processes and service variability modeling. Throughout our paper, we illustrate our ideas with a realistic running example, and also present a proof-of-concept prototype.

**Keywords:** Business Process, SOA, service variability modeling, goal semantics.

## 1 Introduction

In general, a business process is derived based on a specified business goal. However, there are many occasions , where multiple variants of the business process need to be derived to address different usage contexts. At the same time, however, each such variant needs to adhere to the same business goal [7]. We call such variants *goal-preserving* variants of a business process. In existing business process design approaches [11], business processes are usually designed and stored separately from the goals from which they were derived. As a result, derivation of multiple goal-preserving variants becomes a costly and time-consuming exercise. To that end, in this paper, we present a novel approach by which variability modeling and subsequent derivation of goal preserving variants is completely driven

S. Basu et al. (Eds.): ICSOC 2013, LNCS 8274, pp. 499–506, 2013.

by goal decomposition models. In this paper, we assume the following inputs: (a) a goal model (e.g., as depicted in Fig. 2) with goals and associated decomposition of sub goals (AND, OR) represented as a collection of boolean conditions in conjunctive normal form (CNF) [6]; (b) a capability library containing a set of services with semantically annotated *effects* [11]; and (c) a semantically annotated process design created with the composition of such services, said process design adhering to the goal model. Our proposed approach works as follows. First, an initial business process is generated from the goals as per our earlier work [8]. Second, based on the effect annotations, we derive the goal-based *variability analysis model (VAM)* for the services participating in a business process. This model extends our earlier work on service variation modeling [14], to determine all possible variants of the services that adhere to their mapped goals. Third, using the VAM, we generate the required goal-preserving variants for the original process.

This paper is organized as follows. We present related work in Section 2. Section 3 discusses our running example, which is drawn from the insurance domain. In Section 4 we provide some basic definitions and also show how goals can be established as a foundation for variability analysis. We present and discuss our prototype implementation in Section 5, and conclude the paper with suggestions for future work in Section 6.

## 2 Related Work

In Product Line Engineering (PLE) based approaches [2,4], variability of products is systematized in terms of variability identification, modeling, conflict resolution and finally instantiation. But in SOA, a custom developed SOA based application could comprise services and processes developed by different organizations [5]. These services and processes need to be modified for supporting different user contexts, using valid variations that satisfy the corresponding organization goals. Approaches for process variability support such as Provop [9], focus on managing large collections of process variants of a single process model.On similar lines, the citation [13] describes an approach to quantitatively calculate similarity between any two variants of a business process, so that activities such as process reuse, analysis and discovery can be facilitated. This is done via the modeling of process constraints on tasks, such as which tasks should (or should not) execute together. Such methods undoubtedly possess effective variability management techniques, but without alignment between the goal model and business process model. In works such as [10], the process goals are proposed as a collection of tasks with specific input and output parameters, and are matched against existing tasks in a capability library; the matching is accomplished via AI planning techniques. In declarative workflow based approaches [16], constraint satisfaction is employed to address the different types of process flexibility such as differing a modeling decision to a later phase of the process life cycle, accommodating changes to the process design or deviating the process execution from

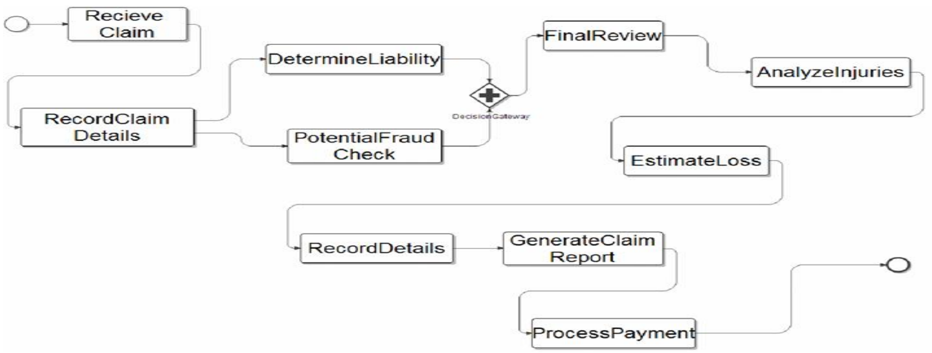


**Fig. 1.** Insurance Claims Process - Solution `Pr1`

modeling time decisions. It requires the constraints to be specified in a declarative language such as DecSerFlow [17], leading to challenges such as management of large collection of process variants in the repository [1]. Our proposed work, on the other hand, provides a more realistic and practical approach wherein we provide the necessary facility for the business analyst to specify process and service goals at a level of abstraction comfortable for him/her. We then provide an approach to decompose the goals into sub-goals until there is an ontological match between them and the semantically annotated effects of services in the capability library. Subsequently the goal-linked process and services are subjected to variability analysis for checking and generating valid variations that continue to preserve the goals but satisfy changing user requirements.

## 3 Running Example

Our running example depicts a goal decomposition model for an organization dealing with different types of insurance claims as illustrated in Fig. 2. The Goal `Process Accident Claims` is primarily decomposed into four mandatory subgoals `Receive Claim`, `Verify Claim`, `Record Claim` and `Analyze Injuries`. Each of these sub goals, contains both mandatory and optional leaf level subgoals that the organization expects to be addressed by different insurance claim business processes. Let us consider an accident claim process `Pr1` as illustrated in Fig. 1. The inputs to this process are the details of the customer requesting the claim, and the details of the claim. The outputs of this process are the acceptance/rejection of the claim, along with the claim amount to be paid to the customer (which will be zero in case of rejection). `Pr1` consists of four major sub-processes - (i) `Record Claim` ($RC$), (ii) `Verify Claim` ($VC$), (iii) `Analyze Injuries` ($AC$) and (iv) `Report` ($RP$). In `Verify Claim` sub-process, the *DetermineLiability (DL)* and *PotentialFraudCheck (PF)* services are first executed in parallel, and then their results are combined and sent to *ClaimInvestigation (CI)* service. A final review of the verified claim is then implemented by *FinalReview (FR)* service. A variant of `Pr1`, adhering to the goal model in Fig. 2

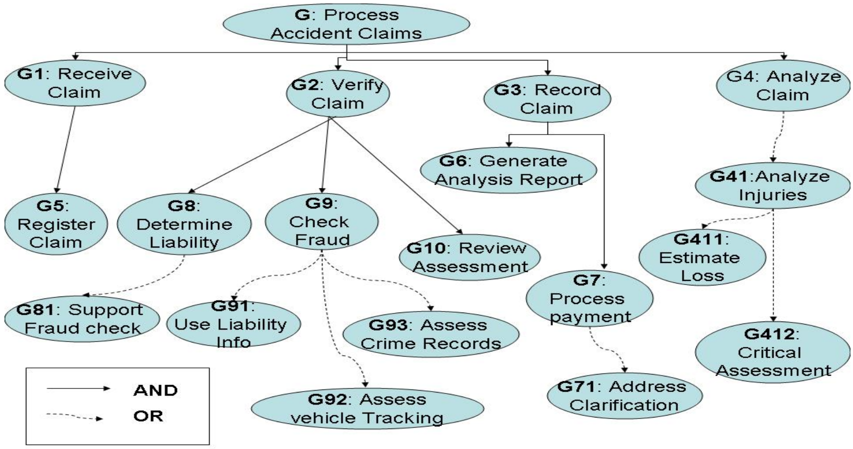


**Fig. 2.** Insurance Claims Process - Goal Model

by satisfying all the mandatory sub-goals (along with a different combination of optional sub-goals) as `Pr1`, could be `Pr2`.The process `Pr2` contains the following differences from `Pr1`: $DL$ and $PF$ services are serialized (satisfying the optional sub-goal `G81`); $PF$ service is modified (satisfying the optional sub-goal `G91`) to also consider the extent of liability from the service $DL$ along with the customer and claim details to determine the possibility of occurrence of fraud. In the rest of this paper, we discuss how our variability analysis model can help generate the goal-preserving variants of `Pr1`, such as `Pr2`.

## 4 A Goal-Oriented Approach to Variability Analysis

In this section, we argue for the centrality of goals in variability analysis.

In our earlier paper [8], we proposed a goal refinement procedure based on the KAOS methodology [3]. With this procedure, we asserted that the set of sub-goals for a goal will achieve the goal (entailment); it will be the smallest set of sub-goals to achieve the goal (minimality); and it will never be incorrect (consistency). In our proposed approach in this paper, we leverage such a refined goal model as illustrated in Fig. 2 to identify the set of minimally required sub-goals to entail the overall process goal.

Let us consider the process `Pr1` from our running example. The goal model for $G$ depicted in Fig. 2 can be expressed in CNF form as $G = G5 \wedge G8 \wedge G9 \wedge G10 \wedge G6 \wedge G7 \wedge G4$. Let us assume the mapping of the services in `Pr1` with the goal model as follows: $Pr1 \vdash G, RC \vdash G5, VC \vdash G2, RP \vdash G3, AI \vdash G4, DL \vdash G8, PF \vdash G9, FR \vdash G10, CI \vdash G411$. Now as an illustration, leveraging these variability mappings, the following variability analysis for the services $DL$ and $PF$ can be established: The service $DL$ can have both interface and implementation level variations that could preserve either the Goals $G8$ or $G81$. Similarly the service $PF$ can have both interface and implementation level variations preserving either of goals $G9$, $G91$, $G92$ and $G93$. For example to support the variant `Pr2` of `Pr1`, we can leverage the variations satisfying $G81$ and $G91$ respectively by the services $DL$ and $PF$. This enables generation of goal-preserving variants.

We therefore establish that a service or process variant is goal preserving only if it eventually adheres to the same goal that the base service adheres to. We discuss the different scenarios in which this goal adherence can be verified. We assume that the existing variant(s) of the service are also available in the capability library and are semantically annotated with end effect scenarios.

Let us consider a service $s_j$ to be a variant of $s_i$. Let $e_i$ and $e_j$ be the corresponding effect annotations of $s_i$ and $s_j$ respectively. Let $e_i = \{c_1, c_2, ..., c_m\}$ and $e_j = \{c_{j1}, c_{j2}, ..., c_{jn}\}$. Let $G_i == \{c_a, c_b, ..., c_k\}$ be a sub goal of $G$. Let the service $s_i$ entail $G_i$. Then the following condition needs to be met: $\{c_a, c_b, ..., c_k\} \in e_i$. This can be illustrated using our running example as follows: consider the Goal $G8$, which contains the following literal: `DetermineLiability = 'yes'`. Now to establish that the service $DL$ satisfies the goal $G8$, we expect the above literals to be part of the end effects of $DL$ such that $DL =$ `DetermineLiability = 'yes'`, `match-past record = 'yes'` . Now to establish that $s_j$ is a goal-preserving variant of $s_i$, one of the three following scenarios have to be met:

1. The service $s_i$ satisfies the AND sub-goal $G_i$ and $G_i$ does not contain a disjunctive clause. In this case, the service $s_j$ can be established as a goal preserving variant of $s_i$, only if the following condition is satisfied: $\{c_a, c_b, ..., c_k\} \in e_j$. This also implies $G_i \in \{e_i \cap e_j\}$. To illustrate, let us assume a variant $DL'$ of service $DL$, which has the end effects as follows: $DL' =$ `DetermineLiability = 'yes'`,`examine-vehicle = 'yes'`. We can establish that $DL'$ also entails the goal $G8$ and hence $DL'$ is a goal preserving variant of $DL$.
   Now for the remaining two scenarios, let us assume an AND sub-goal $G_i$, such that $G_i$ can be expressed as $\{G_{i1} \vee G_{i2}\}$. Let $G_i = \{c_a, c_b, ..., c_k\}$, $G_{i1} = \{c_a, c_b, ..., c_k, c_{k+1}...c_m\}$ and $G_{i2} = \{c_i, c_j, ..., c_k, c_{k+1}...c_n\}$.
2. Let the service $s_i$ satisfy the AND sub-goal $G_i$, which contains a disjunctive clause. Then the service $s_j$ can be established as a goal preserving variant of $s_i$, only if at least one of the following conditions is satisfied: $\{c_i, c_j, ..., c_k\} \in e_j$, $\{c_i, c_j, ..., c_k, c_{k+1}...c_m\} \in e_j$, $\{c_i, c_j, ..., c_k, c_{k+1}...c_m\} \in e_j$. This also implies $G_i \in \{e_i \cap e_j\}$ like the previous condition. This can be again illustrated from the running example using the service $PF$ and the goal $G9$. We observe that $G9$ can be expressed as $\{G91 \vee G92 \vee G93\}$. Let the service $PF =$ `Determine Fraud = 'yes'`, `Spot Investigation = 'yes'` entail the goal $G9 =$ `Determine Fraud = 'yes'`. Let a service $PF' =$ `Determine Fraud = 'yes'`, `Utilize Liability = 'yes'`, `Inspect Vehicle = 'yes'` be a variant of service $PF$, that uses the liability information to check for fraud in process `Pr2`. We see that $PF'$ entails the goal $G91 =$ `Determine Fraud = 'yes'`, `Utilize Liability = 'yes'`, as it satisfies the above condition.
3. Let a service $s_i$ satisfy the OR sub-goal $G_{i1}$. Then the service $s_j$ can be established as a goal preserving variant of $s_i$, only if at least one of the

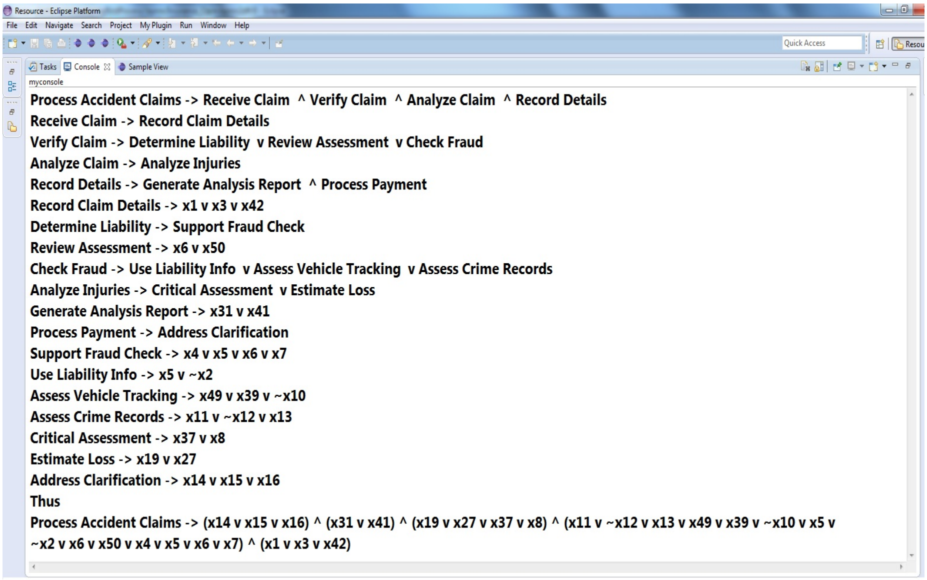


**Fig. 3.** The Generation of Process `Pr1` in CNF

following conditions is satisfied: $\{c_a, c_b, ..., c_k\} \in e_j, \{c_a, c_b, ..., c_k, c_{k+1}...c_m\} \in e_j, \{c_a, c_b, ..., c_k, c_{k+1}...c_n\} \in e_j$ . This implies $\{G_{i1} \vee G_i\} \in \{e_i \cap e_j\}$ . This can be illustrated similarly using the services $PF'$ and $PF$. Now given that $PF'$ preserves the OR sub-goal $G91$, we can establish that $PF$ is a goal preserving variant of $PF'$, as it satisfies $G9$.

## 5 Implementation and Experimentation

For running the experiment to demonstrate goal driven variability analysis, we have developed an Eclipse plugin[1] that helps represent the goal model as a goal-graph. The goal model of the business process depicted in Fig. 2 can be refined as $G = \{G5, G8, G9, G10, G6, G7, G4\}$ using our goal refinement procedure. The CNF form that we have generated from this goal model using our tool is expressed as $G \rightarrow G4 \wedge G5 \wedge G6 \wedge G7 \wedge G8 \wedge G9 \wedge G10$. Let $X = \{xi \mid i = 1, 2...50\}$ be a list of Boolean literals. Each of these sub goals of $G$ are defined as $G4 \rightarrow x1 \vee x3 \vee x42$, $G5 \rightarrow x4 \vee x5 \vee x6 \vee x7$, $G6 \rightarrow x5 \vee \neg x2 \vee \neg x6$, $G7 \rightarrow \neg x10 \vee x49 \vee x39$, $G8 \rightarrow x11 \vee \neg x12 \vee x13$, $G9 \rightarrow x14 \vee x15 \vee x16$ and $G10 \rightarrow \neg X28$. Similarly we represent the process `Pr1` with semantic end effect annotations. We can also observe from the CNF expressions of $G$ and `Pr1`, that `Pr1` satisfies $G$ as illustrated in Fig. 3. In addition to illustrating the implementation with our running example, we also ran an additional experiment with increasing scale of complexity as follows: let us express a process P as

[1] Demo video accessible from `http://variabilitymodelling.wordpress.com`

$P \rightarrow T1 \wedge T2 \wedge T3 \wedge T4 \wedge T5 \wedge T6$. The tasks are defined as : $T1 \rightarrow x1 \wedge x10 \wedge \neg x50$, $T2 \rightarrow x20 \wedge \neg x12 \wedge x39$, $T3 \rightarrow x14 \wedge \neg x50 \wedge x34$, $T4 \rightarrow x7 \wedge \neg x19 \wedge x45$, $T5 \rightarrow \neg x2 \wedge x34 \wedge x15$, $T6 \rightarrow \neg x28$. We ran the CNF form of $G$ using the WinSat tool [15]. We observed that there were 36 different solutions that could satisfy $G$. Now for each service $T1$ through $T6$, we derived the following mappings to the sub-goals of $G$: $T1 \vdash G4$, $T2 \vdash G7$, $T3 \vdash G9$, $T4 \vdash G5$, $T5 \vdash G6$, $T6 \vdash G10$. We can conclude from this that $P \vdash G$. Now for each of the services associated with a sub-goal of $G$, we can perform the variability analysis. For example, for $T1$, the respective goal preserving variants that can be derived are as follows: $T11 \rightarrow x42$, $T12 \rightarrow x3$, $T13 \rightarrow x1$, $T14 \rightarrow x1 \wedge x42$, $T15 \rightarrow x42 \wedge x3$, $T16 \rightarrow x1 \wedge x3$. We can see that each of the variants of $T1$ still satisfy the sub goal $G4$. Similarly we could have goal preserving variants for the other services as well. Similarly, the variants for $T2$ are : $T21 \rightarrow x39 \wedge \neg x12$; $T22 \rightarrow \neg x10 \wedge x11$. And the variants for $T3$ are: $T31 \rightarrow x15 \wedge \neg x27$; $T32 \rightarrow x15 \wedge x16$, while the variants for $T4$ are: $T41 \rightarrow x4 \wedge x5$; $T42 \rightarrow x7$. The variants for T5 are : $T51 \rightarrow \neg x6$; $T52 \rightarrow x5 \wedge \neg x2$. Hence these variants still preserve their mappings with the sub goals and thus the process $P \vdash G$. Now given all the goal preserving variants for each of these services, we could generate goal preserving variants of `P`. This addresses the first objective of our proposed approach, which is goal driven variability analysis. Now suppose we already have existing candidate variants, we can validate whether these variants satisfy the respective goal or not. For example suppose we have an existing variant of T3, $T32 \rightarrow x20 \vee x31$. Since $T3$ entails $G9$ ($G9 \rightarrow x14 \vee x15 \vee x16$), we could infer that $T32$ is not a valid variant of $T3$ as $T32$ could not satisfy the mapping $T3 \vdash G9$. This addresses the second objective of our proposed approach, which is validating existing variants for their goal preserving nature. Let process `P'` be a variant of `P` and be defined as: $P' \rightarrow (x1 \vee x3) \wedge (x20 \vee \neg x12 \vee x39) \wedge (x15 \vee \neg x27) \wedge (x7 \vee \neg x19 \vee x45) \wedge (\neg x2 \vee x34 \vee x15)$. From the list of goal preserving variants and the original services, we can establish that $P' \rightarrow T16 \wedge T2 \wedge T31 \wedge T4 \wedge T5$. Since each of the service variants are goal preserving we can infer that $P' \vdash G$.

## 6 Conclusions

In this paper, we have looked at variability in service-based business processes through a goal-oriented lens. In particular, we have shown how, given a business processs, its goal model, and a set of services (and variants thereof) as part of a capability library, all goal-preserving variants of the business process can be generated. Generating such variants is crucial in cases when organizations need to generate multiple variants of the same business process in order to cater to varying user requirements. We have also presented the conditions under which such goal-preserving variants can be generated. As part of future work, we will address co-evolution of goal models and business process models by integrating ideas from this paper with those from an earlier work [12].

## References


1. Ayora, C., Torres, V., Reichert, M., Weber, B., Pelechano, V.: Towards run-time flexibility for process families: Open issues and research challenges. In: La Rosa, M., Soffer, P. (eds.) BPM 2012 Workshops. LNBIP, vol. 132, pp. 477–488. Springer, Heidelberg (2013)
2. Chen, L., Babar, M.A., Ali, N.: Variability management in software product lines: a systematic review. In: Proceedings of the 13th International Software Product Line Conference, pp. 81–90. Carnegie Mellon University (2009)
3. Darimont, R., van Lamsweerde, A.: Formal refinement patterns for goal-driven requirements elaboration. SIGSOFT Software Engineering Notes 21, 179–190 (1996)
4. Deelstra, S., Sinnema, M., Nijhuis, J., Bosch, J.: Cosvam: A technique for assessing software variability in software product families. In: ICSM, pp. 458–462 (2004)
5. Thomas Erl. Service-oriented Architecture: Concepts, Technology, and Desing. Pearson Education India (2006)
6. Carbonell, J., et al.: Context-based machine translation. In: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas, pp. 19–28 (2006)
7. Ghose, A.K., Koliadis, G.: Auditing business process compliance. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 169–180. Springer, Heidelberg (2007)
8. Ghose, A.K., Narendra, N.C., Ponnalagu, K., Panda, A., Gohad, A.: Goal-driven business process derivation. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) ICSOC 2011. LNCS, vol. 7084, pp. 467–476. Springer, Heidelberg (2011)
9. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: the provop approach. J. Softw. Maint. Evol. 22, 519–546 (2010)
10. Heinrich, B., Bolsinger, M., Bewernik, M.: Automated planning of process models: The construction of exclusive choices. In: ICIS, page Paper 184 (2009)
11. Hinge, K., Ghose, A., Koliadis, G.: Process seer: A tool for semantic effect annotation of business process models. In: Proceedings of the 2009 IEEE International Enterprise Distributed Object Computing Conference, EDOC 2009, pp. 54–63. IEEE Computer Society, Washington, DC (2009)
12. Hoesch-Klohe, K., Ghose, A.K., Dam, H.K.: Maintaining motivation models (in BMM) in the context of a (WSDL-S) service landscape. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) ICSOC 2012. LNCS, vol. 7636, pp. 582–590. Springer, Heidelberg (2012)
13. Lu, R., Sadiq, S.W., Governatori, G.: On managing business processes variants. Data Knowl. Eng. 68(7), 642–664 (2009)
14. Narendra, N.C., Ponnalagu, K.: Towards a variability model for soa-based solutions. In: IEEE SCC, pp. 562–569 (2010)
15. Qasem, M., Prügel-Bennett, A.: Learning the large-scale structure of the max-sat landscape using populations. Trans. Evol. Comp. 14(4), 518–529 (2010)
16. Reichert, M., Weber, B.: Enabling flexibility in process-aware information systems. Springer (2012)
17. van der Aalst, W.M.P., Pesic, M.: DecSerFlow: Towards a truly declarative service flow language. In: Bravetti, M., Núñez, M., Zavattaro, G. (eds.) WS-FM 2006. LNCS, vol. 4184, pp. 1–23. Springer, Heidelberg (2006)