Maintaining Goals of Business Processes during Runtime Reconfigurations

Budoor Allehyani and Stephan Reiff-Marganiec

University of Leicester
University Road, Leicester, LE1 7RH - UK
{baaa2,srm13}@le.ac.uk

Abstract Business processes have been used extensively to describe how a business achieves its goals; more recently they have also been used embedded into workflow or business process engines to drive the processes. However, business procedures and demands change and consequently the processes need to be adapted. We are building on mechanisms to change processes at runtime and investigate the important aspect of ensuring that the process remains true to its original goal. In this paper we outline our framework and focus on the formal assurance of the goal.

Keywords: Workflows, Business processes, Policies, Goal Consistency, Refinement

1 Introduction

Business processes are used to describe how a business achieves a goal and also to drive processes through business process engines. Each business process is described in terms of a workflow, essentially a set of tasks that are conducted in a specific order and the interaction of those tasks with the environment ultimately capturing how a business goal is achieved. Anton [3] defines goals as "high-level objectives of the business, organization or system"; they capture the reasons why a system is needed and guide decisions at various levels within the enterprise. Business process goals can be defined as rule(s) describing the business outcome and are considered at the development phase to ensure they are met at all levels.

Scientific domains have used workflows to structure and execute processes. While in this work the focus is on business processes, we believe that it has merit in the scientific workflow domain, too. For our convenience we use business process interchangeably with workflow in this work.

Today's businesses operate in a very dynamic environment, where change is almost constantly required due to customer demands, legislation and changes to the business' nature (e.g. mergers) as well as the desire to work more efficiently. These changes have implications on how the business operates and hence on the processes describing how the business goals are achieved. Typically making such changes is a matter of redesigning the processes,thus involving business analysts and then updating the software executing the processes. Gorton et al. [8]

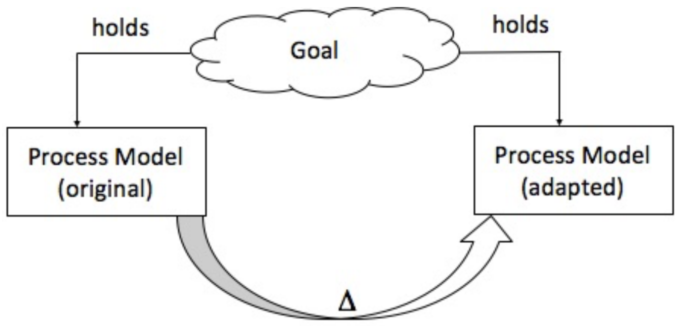

Figure 1. Motivation

presented StPowla which envisioned changes to be made dynamically during process runtime driven by policies describing the rules that one wishes to apply to the process instance. While the previous work captures how the processes are executed and changes are applied it essentially does only provide syntactic guarantees on correctness. However, there is an obvious desire to ensure that the adapted workflow still satisfies its original goal within some sensible range of expectations: no one would want their travel booking process to become one that orders home appliances. More generally, enabling flexibility on workflow systems is a critical challenge in the field of software engineering. Typical issues related to workflow reconfigurations include both syntactical and semantical correctness have been investigated in [5,18,19].

In this work we focus our attention on ensuring consistency in terms of compliance to business goals (note that we only consider functional (hard) goal) . It is of utmost important to ensure that the dynamically reconfigured workflow does what is expected, see Figure 1 where Δ represents the change as defined by the applied policy.

This paper presents our novel contributions (1) the overall framework for goal preserving runtime reconfiguration and (2) the approach to preserving goals.

As an overview, we can consider three different levels for our work, all of which are used at runtime on process instances; note however that obviously the policies and initial processes are specified in a design phase before we execute the process and apply adaptations. Table 1 presents an overview of the levels and their respective inputs and outputs. Specification refers to the original workflow specification including goal and process speciifications, Reconfiguration refers to changing the workflow specification and Verification refers to ensuring that the changed specification does not violate the original goal specification.

The remainder of this paper is structured as follows: Section 2 introduces background work for the formal specification of business processes and their goals as well as StPowla reconfiguration policies. Section 3 presents an overview of our framework and Section 4 presents our example, focuses on the approach ensuring goal compliance. Section 5 presents related work and finally Section 6 concludes on the work and outlines future directions.

Table 1. Overview of runtime levels and languages

	Techniques	Inputs	Outputs
Specification	Wong's prototype tool [20]	BPMN diagram	CSP script
Reconfiguration	Java implementation	CSP + Reconfiguration policies [8]	CSP' script
Verification	Model checking FDR [2]	CSP + CSP' + Semantic properties	Result: Passed/Failed

2 Background

In general, we can look at business processes at several stages of their life cycle; here we assume automatically executed processes only. Typically, we have specifications capturing the design, which are then converted into an executoteable format, which in turn is instantiated and run as and when demanded. If a change is needed this is managed manually at runtime for emergent issues or for more permanent updates by manually redesigning the process. Specifications are supported by a number of languages such as BPMN [1] and there are formal instantiations of such languages in e.g. Petri nets [6] or process algebras such as CSP (Communicating Sequential Process) [20] which allow verification of semantic correctness. We use CSP and its associated tools. Wong [20] provides a formal semantics to (a subset of) BPMN. This work is supported with a prototype tool that converts a BPMN specification into a semantically equivalent CSP specification which then can be further investigated. In their work they focus on deadlock freedom and other generic properties of the process.

Goals of BPMN need to be modelled and formally defined in order to allow for measuring goals achievement [7]. There are plethora of researches in the field of requirement engineering about goal modelling and measuring [10]. KAOS (Keep All Objectives Satisfied) [12] is a requirement specification methodology aimed at supporting requirement elaboration. In KAOS, goal model consists of the strategic goal and its refinement subgoals each mapping to one or several tasks in the process model; these tasks contribute to achieve these subgoals/objectives. Goal modelling in KAOS is declared in two different ways: semi-formal goal structuring model and formal definition in LTL (Linear Temporal Logic). The formal declaration of goals allows for specification in LTL with variant patterns: (1) Achieve goals; the target must eventually occur (desire achievement). (2) Cease goals; there must be a state in the future where the target does not occur (disallow achievement). (3) Maintain goals; the target must hold at all time in the future. (4) Avoid goals; the target must not hold at all time in the future.

In our work we are looking at integrating the change management into the runtime phase by using runtime reconfigurations, which have been proposed in the StPowla [8] framework. Crucially the changes are applied to running instances of a process, so they can make use of data in the instance. As StPowla is meant to operate on running instances of processes the proposed validation needs to fit into the runtime environment. The runtime framework assumes an adapted process execution engine. For simplicity we assume here that we have an engine that can execute BPMN processes directly (this allows us to focus on the main aspects rather than worrying about converting these into some executable formats). The engine is able to pause a process instance and also to make changes to instances.

As the process instance executes it will raise triggers – e.g. at the start of a task which are passed to the policy server (a policy enforcement point) which either returns "no change" allowing the instance to be processed as it is or a specific change action, e.g. the need to insert a task which will lead to updating the process structure of the instance. The action that the policy server demands depends on the policies in the repositories and of course the instance data in the process. The policy server retrieves policies from the policy store, checks for the applicability and then considers the actions to be applied. Once it has determined which actions should be applied the process instance is updated accordingly and would continue executing in its new shape. Through the work presented here an extra phase is added, namely that of checking that the change is appropriate in the sense that it maintains the goal of the original process.

Reichert and Weber [18] analysed typical correctness demands 1) Control flow as well as behavioural correctness, 2) Data flow correctness, 3) Compliance to business rules 4) Instance status compliance and 5) Concurrent change management. Current approaches studied these issues. Semantic correctness in terms of compliance to business rules is addressed in [4] which deals with validating business laws and regulations (compliance rules) as these rules should be met at design as well as reconfiguration levels.

3 Framework for Runtime Goal Assurance – an Overview

As menthioned above, we consider workflow reconfiguration as three level process. The specification and reconfiguration levels was found in [20] and [8] respictively. We integrate these levels into our Java framework and add the verification process. In other words, we implement the reconfiguration policies defined in [8] as Java functions including: i) the insertion of atomic/composite task in parallel/sequence with existing atomic/composite task , ii) the insertion of new branch(s) to existing decision operators and iii) the deletion of atomic/composite task in parallel/sequence with another atomic/composite task. We consider structural correctness in our implementation, e.g. the reconnection of flows when removing atomic existing task as well as inserting new parallel operator when inserting new atomic task in parallel with existing atomic task. For the verification process, we invoke FDR in our framework to check for certain properties (see next section). If the properties hold we allow the change, otherwise we reject it.

4 Approach for Ensuring Goal Compliance

In this work we use the goal concept of KAOS and link it to the process model in order to be able to analyse and reason about the reconfiguration effect. Our verification process benefits from goal modelling patterns (mentioned in section 2) as follows: 1) patterns 1 and 3 allow to ensure the availability of the activities related to the goal when deleting activities from the process and 2) patterns 2 and 4 help to identify the undesired states when inserting new activities.

In KAOS, the strategic goal of the enterprise is refined to different subgoals, see Figure 3, this means that these subgoals are contributing to achieve the main goal. Hence, their representative activities in process model must be available after runtime reconfiguration (as with policy reconfigurations we have the ability to delete activities). Furthermore, the availability of representative activitie(s) for one of these subgoals is not sufficient to ensure the fulfilment of the strategic goal. Note that in the KAOS goal model, these refined goals are connected with AND/OR relations (AND meaning all subgoals are contributing to fulfil the supergoal, OR meaning at least on of the subgoals must be achieved). So, we have two types of check when considering delete policies: 1) the availability of activities contributing to achieving subgoals and 2) the availability of all activities which together fulfil goals. If we consider insert policies we have to check that the new tasks belong to the domain and relate "somehow" to the goal model.

We can also establish a link that preserves/allows these activities from/to changes at policy level. We could call this link a control or management link.

For operational purposes we use CSP tools, so we define the semantic constraints in CSP as well as rules for satisfaction of these constraints. As we want to preserve semantics in the adaptive system, we have the original workflow (source) and reconfigured workflow (target). Our semantic check is based on activities (tasks) type annotation.

So, taking the source (P) and target (P') workflows which are represented in CSP as processes together with the goal specifications G1,....,Gn we wish to check if the property specification refines the goal specification. Such refinement explores the dependency between processes in the process model and the goals in the goal model. Our properties are (informally) defined below:

1. Let X be a type annotation for the tasks which contribute to achieve the goal. All processes of type X in P must be available in P'.
2. Let T be the type annotation for tasks in the relevant domain. All processes in P' must be annotated with one or more type from T.

Here are the steps towards linking goal model to process model: 1. Declare formally the goals/ subgoals using the goal specification patterns, 2. Convert these specifications into CSP specifications, 3. Establish management link between goals and processes. The first step to achieve that is to annotate the contribution activities, 4. Declare our properties, which specify what we want to avoid when change policy affects process model, 5. Define the refinement relation (satisfaction function): Spec |= G; which indicates that the property specification "Spec" satisfies the goal specification in question "G".

Considering our admission example depicted in Figure 2, we suppose that the main (strategic goal) is to assign the right student to the right place. Then, this goal is refined to operational objectives which are related to activities in process model. For example, the activities `Get_GTA`, `Get_Attainment` and `Get_English_Test` are contributing to fulfil the subgoal `RequirementsMet`. Considering this example shows that a block of activities contribute to achieve a

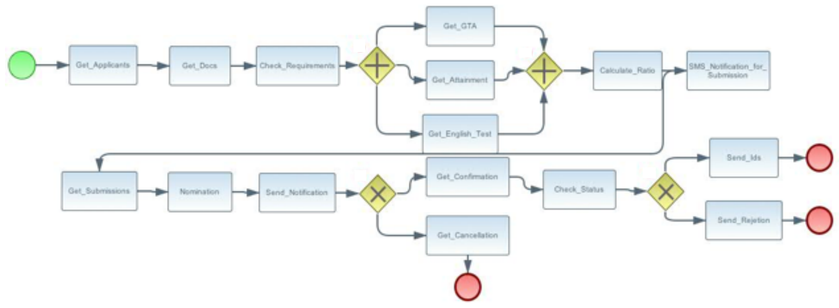

Figure 2. The process model "BPMN" for university admission

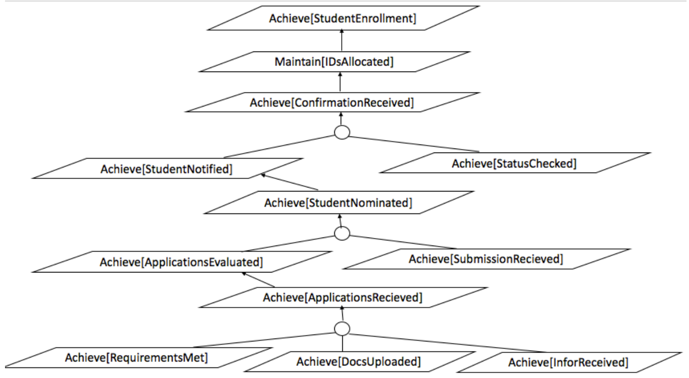
Achieve[StudentEnrollment]
Maintain[IDsAllocated]
Achieve[ConfirmationReceived]
Achieve[StudentNotified]
Achieve[StatusChecked]
Achieve[StudentNominated]
Achieve[ApplicationsEvaluated]
Achieve[SubmissionRecieved]
Achieve[ApplicationsRecieved]
Achieve[RequirementsMet]
Achieve[DocsUploaded]
Achieve[InforReceived]

Figure 3. KAOS goal model for university admission

single subgoal.Therefore, we need a CSP assertions that check the fulfilment of each subgoals to ensure the overall achievement of the strategic goal. For verification we consider CSP trace and refusal trace refinement [16] as we are interesting in the availability of events. A process trace is a sequence of all events the process can execute. This helps us to formulate our assertions as we check the trace for certain (annotated) processes.

5 Related Work

In the literature, workflow adaptation is a twofold issue: 1) providing flexible workflows by adding/deleting/skipping tasks, changing the order of tasks and rolling back failed tasks and 2) ensuring correctness and consistency; it is inevitable to ensure the robustness of adaptive workflows. The robustness of workflow systems includes correct control and data flows, consistent behaviour and consistent semantics. Change can take place at different levels as classified in [9]. Most of the current approaches consider process change at design time handling the change either manually [17], or semi-automatically [14], [15]. StPowla [8] provides a promising solution towards flexible workflows as it handles changes (online) on running instances and automatically through reconfiguration policies. StPowla considers two types of policies, refinement and reconfiguration policies. Briefly, refinement policies specify requirements on the service that can be chosen to execute a task while reconfiguration policies make changes to the workflow structure. It is the latter that are the focus of this work. Reconfiguration policies allow at the most fundamental level for insertion or deletion of tasks in the process, which can easily be extended to inserting/deleting sub-processes and changing operators.

Koliadis and Ghose [11] have studied how the process model is affected when adapting the goal model by establishing (traceability and satisfaction) links between process model (in BPMN) and goal model (using KAOS). We study the effect of process reconfigurations but when adapting the process model.

Ly et al. [13] provide a definition for semantic constraints and their satisfaction in adaptive Process Management Systems (PMS). Their approach based on integrating application knowledge into adaptive PMS, which allows to define semantic constraints based on this knowledge. This work differs from our work as we consider consistency in terms of adhering to the main goal at an abstract level, while they address it at data level considering tasks incompatibility.

We believe that the criteria from [18] are insufficient to ensure workflow robustness as they do not guarantee that the workflow will still adhere to its original specification. We add 6) compliance to business goal as an important semantic constraint when adapting workflows as an additional requirement and the work presented here is a step towards allowing assurance of that.

6 Conclusion and Outlook

Much work has been considering adaptation of workflows; typically this is done manually and not at runtime of process instances. This has the disadvantage that it cannot react to data in the instance and also that it usually requires human intervention. We have previously presented an approach [8] that can dynamically adapt running instances – however that leaves a problem of ensuring that the process still works towards the business goal. In this work we have presented our approach for ensuring such goal compliance for runtime reconfigurations based on formal refinement checking. We are considering semantic consistency against high level specification, taking into account structural correctness. Nevertheless, workflows are knowledge intensive systems that carry and exchange data during execution and it is of utmost important to guarantee data flow correctness but it is outside the scope of this paper. Note that there is mapping from BPEL, the BPMN execution language, to CSP [21] and it is therefore possible to model check data properties through FDR. Future work will investigate the formulation of more generic assertions that can be validated automatically.

References

1. Object management group business process model and notation, `http://www.bpmn.org`
2. Oxford university computing laboratory, `http://www.cs.ox.ac.uk/projects/concurrency-tools/`
3. Antón, A., et al.: Goal-based requirements analysis. In: Requirements Engineering, 1996., Proceedings of the Second International Conference on. pp. 136–144. IEEE (1996)
4. Awad, A., Decker, G., Weske, M.: Efficient compliance checking using BPMN-Q and temporal logic. In: Proceedings of the 6th International Conference on Business Process Management. pp. 326–341. BPM '08, Springer-Verlag, Berlin, Heidelberg (2008)
5. Casati, F., Ceri, S., Pernici, B., Pozzi, G.: Workflow evolution. Data Knowl. Eng. 24(3), 211–238 (Jan 1998)
6. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in bpmn. Inf. Softw. Technol. 50(12), 1281–1294 (2008)
7. Edirisuriya, A., Zdravkovic, J.: Goal support towards business processes modelling. In: Innovations in Information Technology, 2008. IIT 2008. International Conference on. pp. 208–212. IEEE (2008)
8. Gorton, S.: Policy-driven Reconfiguration of Service-targeted Business Processes. PhD thesis, University of Leicester (2011)
9. Han, Y., A. Sheth, A., Bussler, C.: A taxonomy of adaptive workflow management. Workshop of the 1998 ACM Conference on Computer Supported Cooperative Work (1998)
10. Kavakli, E.: Goal-driven requirements engineering: modelling and guidance. Ph.D. thesis, University of Manchester (1999)
11. Koliadis, G., Ghose, A.: Relating business process models to goal-oriented requirements models in kaos. In: Advances in Knowledge Acquisition and Management, pp. 25–39. Springer (2006)
12. Lapouchnian, A.: Goal-oriented requirements engineering: An overview of the current research. University of Toronto p. 32 (2005)
13. Ly, L.T., Rinderle, S., Dadam, P.: Semantic correctness in adaptive process management systems. Springer (2006)
14. Müller, R., Greiner, U., Rahm, E.: Agent work: A workflow system supporting rule-based workflow adaptation. Data Knowl. Eng. 51(2), 223–256 (Nov 2004)
15. PANG, S., Yin, L., Hua, H., Ch, L.: A model for dynamic business processes and process changes. Chinese Journal of Electronics 20(4) (2011)
16. Phillips, I.: Refusal testing. Theoretical Computer Science 50(3), 241 – 284 (1987)
17. Reichert, M., Dadam, P.: Adeptflex—supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems 10(2), 93–129 (1998)
18. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies. Springer Science & Business Media (2012)
19. Rinderle, S., Reichert, M., Dadam, P.: Correctness criteria for dynamic changes in workflow systems: A survey. Data Knowl. Eng. 50(1), 9–34 (Jul 2004)
20. Wong, P.: Formalisations and Applications of Business Process Modelling Notation. PhD thesis, University of Oxford (2011)
21. Yeung, W.: Csp-based verification for web service orchestration and choreography. Simulation 83(1), 65–74 (Jan 2007)