Monitoring, Measuring and Controlling Workflow Applications: A Domain-specific Language Approach *

Oscar González[1,2]. Advisor: Rubby Casallas[1]

[1] Universidad de los Andes, TICSw Group, Cra. 1 N○ 18A 10, Bogotá, Colombia
[2] Vrije Universiteit Brussel, Software Languages lab, Pleinlaan 2, 1050 Brussel, Belgium
{o-gonza1,rcasalla}@uniandes.edu.co

Abstract. In workflow management systems, analysis concerns related to monitoring, measurement, and control aims at identifying potential improvements for workflow applications. A workflow application automates a business process in a control-flow of activities defined at a conceptual level, followed by their implementation in a particular workflow language and engine. Despite the wide range of techniques developed to monitor and analyze workflow applications continuously, the specification of the analysis concerns is done at the workflow implementation level, producing entangled code that is detrimental to their maintainability. The purpose of this research is to offer a solution to automate the implementation of analysis concerns in workflow applications by increasing the level of abstraction in the analysis concerns specification. First, we created of a domain-specific language to specify analysis concerns related to a workflow application in an uniform and technology independent way. Second, it shows a strategy to assist developers to enhance a given workflow technology to support the automated implementation of analysis concerns and their composition with a workflow application. Thus, given a workflow specification and its analysis concerns, they are automatically integrated producing an executable workflow application.

1 Context

Workflow management systems (WFMS) are used to model, implement, execute, and analyze workflow applications. Workflow applications facilitate the automation of processes in an organization, by defining a control flow of activities that manage organizational data [24]. From a traditional point of view, the life cycle of a workflow application starts by modeling it using a high-level notation such as Business Process Modeling Notation (BPMN) [13]. BPMN is currently the de-facto standard and offers a technology independent notation to model

* Supported by the Flemish Interuniversity Council (VLIR) funded CARAMELOS project and the "Instituto Colombiano para el Desarrollo de la Ciencia y la Tecnología COLCIENCIAS

processes. Then, BPMN workflow diagrams are implemented into executable workflow applications by using a workflow programming language (*e.g.,* XPDL [26], BPEL [15], JPDL [16]) to specify the execution semantics of the activities. Subsequently, workflow instances are enacted by workflow engines (*e.g.,* Apache ODE, jBPM).

A crucial phase in the life cycle of workflow applications, in which we focus our research efforts, is workflow analysis. Workflow analysis aims at identifying potential improvements for workflow applications by recovering detailed feedback during their execution. Workflow analysis has multiple application fields such as verification or validation of workflow models, automatic generation of improved workflow models, predictions, analysis done *a posteriori*, and interactive monitoring. However, We consider an accurate workflow improvement by implementing analysis concerns related to monitoring, measuring and controlling continuously the behavior of workflow applications and the data used by its activities. A continuous analysis means combining analysis concerns with workflow code to access monitoring and measurement information on real time.

2 Problem

Despite the wide range of tools and techniques developed so far to analyze workflow applications continuously [19] [18], workflow developers have to manually intervene in the implementation of the workflow application to support the specification of analysis concerns. Therefore, we address the following three associated problems:

Current workflow analysis solutions have limited expressiveness. There is a lack of abstractions in the workflow technologies, since they do not provide primitive built-in analysis constructs. As a result, the implementation of analysis concerns is done in terms of the (low-level) executable workflow implementation, resulting in crosscutting and entangled code in the workflow application. Crosscutting issues are present since one particular analysis requirement can involve several artifacts of the workflow implementation. In these multiple artifacts it is necessary to implement the analysis requirement by indicating a) what to observe in the workflow application, b) the data that is required to measure the workflow application, and c) the actions that need to be accomplished.

In addition, the implementation of analysis concerns is entangled (mixed) with and tied to the code statements of the workflow implementation. Since the implementation of the analysis concerns is done manually by a programmer, this is detrimental to the comprehensibility and maintainability of the workflow and analysis implementation. Thus, once an analysis requirement changes, its implementation has to be repeated or adapted in multiple artifacts of the workflow implementation. In summary, we can state that analysis is treated as a 2nd class citizen by existing workflow solutions.

Current workflow analysis solutions are tied to particular workflow platforms. Typically, specifying code to analyze a workflow application requires knowledge of the technology used to implement it. The inherent use of

proprietary languages (not tuned to analysis) complicates the work of workflow analysts to encode analysis concerns in the implementation of the workflow application. Thus, an analysis specification requires experts in the executable workflow language (*e.g.,* JPDL, BPEL) and workflow engine (*e.g.,* jBPM, Apache ODE). In summary, we can state that analysis concerns are not provided in a uniform and technology independent way.

Current workflow analysis solutions provide no support for domain-specific workflow analysis. Workflow monitoring solutions typically only allow workflow developers specifying analysis concerns in terms of workflow engine execution stack (*e.g.,* the time a workflow is running, the current workflow execution state, the number of instances of a process, the utilization of workflow resources) but not in terms of information that is specific to the particular domain of the workflow application (*e.g.,* Insurance, Banking, Customer Support). In summary, we can state that analysis concerns are generic but not specialized in the particular domain the workflow is modeling.

3 Research Goals and Approach

The main goal of this research is to offer a solution to automate the implementation of analysis concerns in workflow applications by increasing the level of abstraction in the analysis specifications at a conceptual level.

We defined a number of desirable solutions in our approach for addressing the problems to implement workflow analysis concerns. We propose a workflow analysis approach providing a) workflow language and engine independence in the analysis specification to specify analysis concerns in an uniform and technology independent way, b) domain-specific analysis (data-centric) to tackle the lack of expressiveness, and c) analysis concerns modularization and automatic composition with the workflow implementation to automate the analysis implementation.

We propose a strategy to automate the implementation of workflow analysis concerns by separating the specification from its implementation and from the implementation of the workflow application. The overall approach we adopt is the creation of a domain-specific language to specify generic and domain-specific analysis concerns related to a workflow application. This analysis specification is consequently integrated with the existing workflow implementation in an automated fashion.

The main elements of our approach are the workflow analysis specification, the workflow analysis implementation, and the workflow analysis execution.

Workflow Analysis Specification. Our domain-specific language is designed to specify analysis concerns comprised of three activities: Monitoring, Measurement, and Control (MMC) [9] [11]. The *Monitoring activity* uses *monitoring subjects* to intercept certain *monitoring interactions* in the workflow execution for capturing *monitoring events* with information relevant to analyze the workflow application. The *Measurement activity* uses *measurement actions* to specify new *measurement data* and to correlate process and measurement infor-

mation to measure the workflow application. The *Control activity* uses *evaluation operations* to detect special behaviors over the workflow and measurement data and to specify *control actions* to be taken when the monitoring events are triggered. Analysis concerns are specified independently of any workflow technology and in a modularized way since they are declared externally and in terms of the workflow specification.

Since our workflow analysis approach looks for supporting domain-specific analysis specifications, a complete description of the data entities used by the workflow application and those ones required to measure it have to be provided. We defined an approach to modeling data entities on workflow diagrams and on the associated monitoring rules on that workflow. Data modeling corresponds to a projection of the underlying data values that are made available to a client for visibility purposes. This projection facilitates the usage and understanding of data entities and the maintainability of this projection.

We use three models to describe the data used by a workflow application and its analysis concerns: data types model, process variables model, and measurement variables model. The *Data Types* model represents the structure of the data and the data interchange format between the workflow domain and analysis concerns. *Process variables* are declared to complement the data entities specifications by describing the data entities used by the flow entities in the workflow application. Thus, the monitoring of relevant workflow data can be specified and captured selectively and explicitly in order to reduce the complexity of analysis activities at processing large volumes of workflow data. The declaration of *measurement variables* facilitates specifying generic and domain-specific metrics to analyze the process. This data-centric specification facilitates workflow analysts to specify domain-specific analysis concerns (*e.g.,* loans requested by a specific person) driven by the domain the workflow is modeling (*e.g.,* Banking).

We use Model-Driven Engineering (MDE) to raise the level of abstraction in the analysis specifications and to decrease the complexity required to implement analysis concerns. MDE Technology uses language definitions to supply the effective expression and creation of complex platforms at a high level of abstraction [23]. MDE defines the structure and behavior of applications within a particular domain using *models* and *model transformations*. These domain-specific models are described using metamodels, which define the relationships between domain concepts. On the other hand, model transformations are programs that transform input models by using transformation rules into multiple artifacts such as text (*e.g.,* source code, descriptors) or another model representation.

We have defined a metamodel for creating workflow analysis models and an editor to declare analysis specifications. Specifications written in a domain-specific language can automatically generate system families by using generative software development [5].

Workflow Analysis Implementation. We propose a workflow analysis generation process to support the automated implementation of analysis concerns. Our workflow analysis implementation process transforms analysis specifications into executable analysis code. The executable workflow implementation

and the generated executable analysis code are automatically weaved to generate an *instrumented executable workflow implementation* enriched with analysis concerns, which can be executed in a workflow engine [10].

We use multiple generative programming approaches to automate the implementation of analysis concerns into diverse workflow platforms. Model transformations in an Model-Driven Engineering (MDE) context [23] are used to translate analysis specifications declared with our DSL into executable workflow language code, which is encapsulated using aspect-oriented programming (AOP) technology. This mechanism facilitates introducing the crosscutting analysis behavior into the workflow implementation in a modularized way [17]. Therefore, the code that implements the analysis concerns and connects them with the workflow implementation can be identified to enhance its maintainability [8]. Thus, if the workflow specification changes, the analysis concerns implementation can be re-generated a composed with the new workflow implementation without loosing the analysis specification and reducing the time to implement analysis concerns. We use annotations to add non-functional information such as an explicit data management in the workflow implementation for analysis purposes. This is by intercepting events over the information inside the flow entities (*e.g.,* activities, gateways) and not only about events over the workflow execution.

Our workflow analysis generation process defines a strategy to assist developers by using multiple generative approaches to enhance a given workflow technology to support the automated implementation of analysis concerns and their composition with a workflow application.

Workflow Analysis Execution The resulting code artifacts correspond to an executable workflow implementation instrumented with analysis concerns that can be executed in a workflow engine (*e.g.,* Appahe ODE, jBPM) supporting the adopted workflow language (*e.g.,* BPEL, JPDL).

While workflow instances are in execution, the analysis concerns interact with a) the workflow engine to access execution information, b) a workflow data system to retrieve the workflow application information, c) a measurement data system to manage analysis information, and d) a dashboard to visualize analysis concerns (*e.g.,* metrics, alarms) for identifying potential workflow improvements.

Our approach records relevant information regarding workflow entities, monitoring events, and analysis information into a centralized measurement data model. We do not consider a data integration solution since it would be heavyweight.

We designed and implemented an interactive dashboard to alert workflow analysts of special analysis behaviors defined over workflow applications. This web-based user interface facilitates be aware of the current analysis rule situation so it could be defined an effective way to address the situation to program a workflow improvement.

4 Related Work

Several commercial BPM products offer solutions for business activity monitoring [25] [19] [18]. Typically, these solutions offer rich dashboards and extract the information from audit trails, in which workflow metrics are added to the workflow architectures for analysis. In contrast, we propose a high-level language to describe, independently of specific implementation languages, the domain entities (flow and data) we want to monitor, the custom measures we want to build, and the control actions we want to apply over this information. We include a data association model in our approach to declare explicitly the data used by the workflow models for easing domain-specific analysis specifications. Furthermore, we provide a solution to automate the implementation of analysis specifications into an existing workflow implementation.

Certain approaches use data mining and data warehousing techniques to recover the workflow execution information and to discover structural patterns (*i.e.,* decision trees) for identifying the characteristics that contribute to determine the value of a metric [12]. In our approach, there is an explicit definition of the relations between existing metrics and workflow data to build new domain-specific metrics. This information is captured and analyzed during the operation of a workflow application. Although these approaches do not address any of our problems, our approach could benefit from the information provided by these approaches since they can provide values for new measures that were not specified and implemented for an operational workflow application.

Other works [14] [6] [12] [21] introduce the idea to perform business process management using taxonomies and ontologies to capture semantic aspects using process mining techniques. Similar to those approaches we also acknowledge the need to consider analysis capabilities at the knowledge level. In addition, we introduce the idea of a language dedicated to workflow analysis, using a generative approach to automate the implementation of analysis concerns into particular workflow languages and engines. We provide a practical way to be able to use the semantic knowledge for workflow analysis. We complement the high-level process models with the description of process data and the interactions with workflow entities required to perform a domain-specific analysis (data-centric analysis). Our approach facilitates to perform a similar performance analysis (measures) to that ones using a mining approach. Our approach can be used to support process mining approaches by specializing the control action *trace* to provide the facilities necessary for creating event logs with structured analysis information.

Pedrinaci et.al. [22] presents an approach for supporting strategy-driven business process analysis by using an ontology to capture strategic concerns. This approach is similar to ours in the sense that analysis concerns (or strategic concerns in terms of this strategy-driven approach) are declared in a high-level of abstraction (conceptual level). In our approach, we complement this conceptual analysis specification defining explicitly the execution semantics of analysis concerns by using model transformations to generate code into a specific workflow platform. In addition, we offer a mechanism to declare complex measurement

data to facilitate the specification of domain-specific metrics (data-driven metrics) and a control activity to declare control actions over the measurement data.

The authors in [4] present business provenance as a technology for tracking and correlating relevant aspects of business operations (end-to-end operations). Thus, provenance technologies help understanding what actually happened during the life cycle of a process. This technology is related with Business Activity Monitoring (BAM) since business provenance traces operations to extract relevant information. Whereas business provenance supports continuos monitoring using rules that enrich the workflow application (provenance graph), we offer an approach for an explicit specification of analysis rules, which keep external and modularized with respect to the workflow implementation. Business provenance, as we do, treats data as first class entities since they outrun the life cycle of the process. That is why we focus on supporting events and relations in terms of data interactions that are important for a particular analysis specification. This business provenance technology could complement our approach by adding a historical perspective for enabling root cause analysis by providing time series navigational views (*e.g.,* events, user's actions) over the execution of workflows. In this way, it would be possible to realize when a particular behavior occurs, the resource responsible for it, and why this took place.

In [7], Foster et.al. discuss how to model and implement service state and the associated interactions in that state. The state is related to the data values associated with a service that persist across interactions, whereas the interactions are related to name state, access that state, modify that state, and destroy it. Although this work is focussed in a Web Services domain, we tackle a similar problem since we acknowledge the necessity to explicitly define and implement data variables (workflow and measurement) and their interactions but in a workflow application context. In addition, we discuss the issue about how to intercept CRUD (create, read, update, delete) interactions (events) on the data entities to support domain-specific monitoring operations.

5 Results

We have tested our approach within two different workflow applications, which are defined in two different workflow platforms. The first workflow application corresponds to a trouble ticket process [20], which concerns the processing of problems in a software product. We also applied our workflow analysis approach to specify and implement analysis concerns into a loan process [15]. This workflow application corresponds to a typical and real life process used to manage loans in a financial company.

We performed multiple experiments to automate the implementation of analysis concerns and their composition with these workflow applications by using our domain-specific language and our generation process. The analysis concerns specified with our domain-specific language can be automatically integrated with workflow applications automated in BPEL and JPDL. On the one hand, the analysis concerns specified for a workflow application automated in BPEL can be

automatically translated into aspects code by using the aspect language named Padus [3], which can be integrated with the BPEL code. On the other hand, analysis concerns specified for a workflow application automated in JPDL can be automatically translated into aspects code in the AspectJ [2] language, which can be integrated with java code.

We contribute to the workflow analysis field in several ways such a) introducing the idea of a language dedicated to the analysis of workflow applications, b) proposing a model-based approach to specify analysis concerns independently of specific platforms, c) defining new language abstractions for data-centric analysis specifications that ease the specification of domain-specific metrics, d) defining a dedicated language for the definition of data entities and their association with flow entities, e) defining and implementing a strategy to automate the implementation of analysis concerns into diverse workflow platforms by using multiple generative programming approaches, and d) combining and adopting both levels of abstraction (conceptual and implementation) in our approach to provide a entire solution for workflow analysis.

Since the monitoring and analysis specifications are independent of particular workflow platforms, different workflow platforms can be targeted and the workflow analysts can be involved in the specification. Because of the specification of analysis concerns is separated from the workflow specification, the analysis concerns can be easily identified and adapted. If the workflow specification changes, the analysis concerns can be re-generated and composed with the new workflow implementation. Finally, our approach to involve the data managed by the workflow applications in the specification of analysis concerns facilitates the evaluation of performance measurements in terms of the specific domain the workflow application is modeling.

References

1. W. Abramowicz, editor. *Business Information Systems, 12th International Conference, BIS 2009, Poznan, Poland, April 27-29, 2009. Proceedings*, volume 21 of *Lecture Notes in Business Information Processing*. Springer, 2009.
2. AspectJ Team. The aspectj programming guide. http://www.eclipse.org/aspectj.
3. M. Braem, K. Verlaenen, N. Joncheere, W. Vanderperren, R. V. D. Straeten, E. Truyen, W. Joosen, and V. Jonckers. Isolating process-level concerns using padus. In S. Dustdar, J. L. Fiadeiro, and A. P. Sheth, editors, *Business Process Management*, volume 4102 of *Lecture Notes in Computer Science*, pages 113–128. Springer, 2006.
4. F. Curbera, Y. N. Doganata, A. Martens, N. Mukhi, and A. Slominski. Business provenance - a technology to increase traceability of end-to-end operations. In R. Meersman and Z. Tari, editors, *OTM Conferences (1)*, volume 5331 of *Lecture Notes in Computer Science*, pages 100–119. Springer, 2008.
5. K. Czarnecki. Overview of generative software development. In J.-P. Banâtre, P. Fradet, J.-L. Giavitto, and O. Michel, editors, *UPP*, volume 3566 of *Lecture Notes in Computer Science*, pages 326–341. Springer, 2004.
6. A. K. A. de Medeiros, C. Pedrinaci, W. M. P. van der Aalst, J. Domingue, M. Song, A. Rozinat, B. Norton, and L. Cabral. An outlook on semantic business process

mining and monitoring. In R. Meersman, Z. Tari, and P. Herrero, editors, *OTM Workshops (2)*, volume 4806 of *Lecture Notes in Computer Science*, pages 1244–1255. Springer, 2007.
7. I. T. Foster, S. Parastatidis, P. Watson, and M. Mckeown. How do i model state?: Let me count the ways. *Commun. ACM*, 51(9):34–41, 2008.
8. O. González, R. Casallas, and D. Deridder. Modularizing monitoring rules in business processes models. In R. Meersman, Z. Tari, and P. Herrero, editors, *OTM Workshops*, volume 5333 of *Lecture Notes in Computer Science*, pages 22–23. Springer, 2008.
9. O. González, R. Casallas, and D. Deridder. Mmc-bpm: A domain-specific language for business processes analysis. In Abramowicz [1], pages 157–168.
10. O. González, R. Casallas, and D. Deridder. Automating the implementation of analysis concerns in workflow applications. In *24th IEEE/ACM International Conference on Automated Software Engineering*, submitted for publication.
11. O. González, R. Casallas, D. Deridder, and V. Jonckers. A high-level domain-specific language for business processes monitoring and measurement. In *6th Belgian-Netherlands Software Evolution workshop (BENEVOL 2007)*, 2007.
12. D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan. Business process intelligence. *Computers in Industry*, 16(3):321–343, April 2004.
13. O. M. Group. Business process modeling notation specification - final adopted specification, February 2006. http://www.bpmn.org.
14. M. Hepp, F. Leymann, J. Domingue, A. Wahler, and D. Fensel. Semantic business process management: A vision towards using semantic web services for business process management. In F. C. M. Lau, H. Lei, X. Meng, and M. Wang, editors, *ICEBE*, pages 535–540. IEEE Computer Society, 2005.
15. IBM. Business process execution language for web services, July 2002. http://www.ibm.com/developerworks/library/specification/ws-bpel/.
16. JPDL homepage. http://www.jboss.org/jbossjbpm/jpdl/.
17. G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J.-M. Loingtier, and J. Irwin. Aspect-oriented programming. In *ECOOP*, pages 220–242, 1997.
18. C. Lau, S. Peddle, and S. Yang. Gathering monitoring metrics to analyze your business process, December 2007. IBM, http://www.ibm.com/us/.
19. D. Miers, P. Harmon, and C. Hall. The 2007 bpm suites report – a detailed analysis of bpm suites version 2.1, 2007. Business Process Trends,http://www.bptrends.com/.
20. Nortel. Workflow scenario: Trouble ticket. Technical report, March 1998.
21. C. Pedrinaci, J. Domingue, C. Brelage, T. van Lessen, D. Karastoyanova, and F. Leymann. Semantic business process management: Scaling up the management of business processes. In *ICSC*, pages 546–553. IEEE Computer Society, 2008.
22. C. Pedrinaci, I. Markovic, F. Hasibether, and J. Domingue. Strategy-driven business process analysis. In Abramowicz [1], pages 169–180.
23. D. C. Schmidt. Guest editor's introduction: Model-driven engineering. *IEEE Computer*, 39(2):25–31, 2006.
24. W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske. Business process management: A survey. In *Business Process Management*, pages 1–12, 2003.
25. M. von den Driesch and T. Blickle. *Operational, Tool-Supported Corporate Performance Management with the ARIS Process Performance Manager*. Aris in practice edition, 2006.
26. XPDL homepage. http://www.wfmc.org/xpdl.html.