Using MiniZnMASC Middleware with Different Algorithms for Business-Driven Adaptation of Business Processes

Qinghua Lu¹,², Vladimir Tosic²,¹

¹ School of Computer Science & Engineering, University of New South Wales, Sydney, Australia
² Business Adaptation and Interoperation Project, Software Systems, NICTA*, Sydney, Australia

Qinghua.Lu@nicta.com.au, vladat@computer.org

Abstract—Runtime adaptation of business processes and their IT system implementations to changes can usually be done in several ways. MiniZnMASC middleware makes adaptation decisions that maximize business value while satisfying all given constraints. All necessary information about alternative adaptations and their business metrics are specified as policies in WS-Policy4MASC. Using an example loan application business process, we demonstrate how MiniZnMASC supports 4 different autonomic business-driven decision making algorithms for adaptation.

Keywords— business-driven IT management; autonomic computing; dynamic adaptation; decision making; business process management; constraint programming; REST

I. INTRODUCTION

When changes occur in running business processes, adaptation can usually be done in several ways and then advanced decision making is needed to determine how to proceed with the adaptation. In the business-driven IT management (BDIM) [1] research area, decisions are made based on mappings between business and technical IT metrics and are performed to maximize total business value for the enterprise. Autonomic computing [2] is an approach where IT systems self-manage themselves using configurable policies, with minimal human intervention. A policy formally specifies a collection of high-level, implementation-independent, operation and management goals and/or rules. Autonomic BDIM is the intersection area of autonomic computing and BDIM, which adds processing of business metrics to decision-making components of autonomic systems.

Our MiniZnMASC middleware [3] implements different autonomic BDIM decision making algorithms for business process management. MiniZnMASC uses the WS-Policy4MASC policy language [4-5] for description of various adaptations and all information necessary for decision making. WSPolicy4MASC extends the WS-Policy industry standard and defines 5 types of WS-Policy policy assertions: 1) goal policy assertions prescribe conditions to be met; 2) action policy assertions list adaptation actions; 3) utility policy assertions contain financial and non-financial business metrics for particular situations; 4) probability policy assertions specify probabilities of occurrence; and 5) meta-policy assertions describe which business value metrics are important for adaptation decisions when several action policy assertions can be applied but only 1 has to be chosen. When events occur in the managed business process, its IT implementation or its environment and other conditions specified in relevant WS-Policy4MASC policy assertions are satisfied, each triggered action policy assertion represents an adaptation option. If the number of such adaptation options is more than 1, the Policy Conflict Resolution module of MiniZnMASC selects the one that is best from the business viewpoint, using information in WS-Policy4MASC policy assertions and a decision making algorithm. This paper summarizes our demonstration of how MiniZnMASC supports various autonomic business-driven decision making algorithms. These algorithms differentiate MiniZnMASC from recent middleware for adaptation of business processes and their service-oriented implementations, such as [6-10]. Due to the modular design of MiniZnMASC [3] these and other algorithms can be substituted easily.

II. FOUR ALGORITHMS FOR BUSINESS-DRIVEN MANAGEMENT

A. The basic BDIM algorithm

Our basic BDIM conflict resolution algorithm described in [4-5] first loops through all conflicting adaptation options to check whether each of them satisfies all given constraints and, if yes, calculates sum of all relevant business value metrics (BVMs) for this adaptation option. The algorithm calculates summary business value metrics for each adaptation option represented as an action policy assertion, based on WS-Policy4MASC utility and probability policy assertions. Meta-policy assertions specify which types of business metrics are relevant for decision making in particular situations. Adaptation options that satisfy all constraints are added to the list, along with their summary BVM. If none of the conflicting adaptation options satisfies the constraints, the resulting list is empty and an exception is thrown. The adaptation options in the list are ordered based on the decreasing value of their summary BVM, so the first adaptation option in the list has the highest summary BVM and satisfies all given constraints. This adaptation option is returned as the adaptation action to be executed.

* NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centres of Excellence program.
B. The extended algorithm using the Business Motivation Model (BMM) information

This extended algorithm uses information about longer-term BMM [11] business ends and means, added in the newest extension of WS-Policy4MASC. When the summary BVM is calculated for each adaptation alternative, this extended decision making algorithm adds weighted contribution towards the overall BMM business vision. The algorithm for calculation of the contribution to the overall business vision traverses the AND/OR hierarchy of BMM ends and means, calculates the summary business value for each hierarchy node, and applies utility contribution weights and occurrence probabilities. This algorithm enables choosing an adaptation alternative that might not be the best in the short-term but is the best one when longer-term business considerations are taken into account.

C. The algorithm for concurrent adaptation of multiple instances

This algorithm can concurrently make different adaptation decisions for different classes of business process instances in a way that achieves globally optimal total business value while satisfying all given constraints. Here, a “class of instance” is a group of business process instances that share a combination of characteristics that warrants adaptation in the same way. The most important among these characteristics are: the implemented business process type, the executed Web service composition, the current position/state within the running Web service composition, the class of consumer, and other contractual obligations to the consumer. The number of running business process instances in each class also affects which adaptation options are selected. It changes during runtime as instances are started or stopped, so a class of instance can be adapted in a different way when the number of instances in any class changes. To find the globally optimal set of adaptation decisions, MiniZnMASC uses models in the powerful constraint programming language MiniZinc [12], which has a free efficient constraint solver we integrated into MiniZnMASC.

D. The algorithm for adaptation of RESTful business processes

In RESTful business process [13], in decision making points several possible “next steps” process fragments are provided from which one has to be chosen. We extended MiniZnMASC with new decision algorithms determining which process fragment to execute for particular users in RESTful business process systems. These selections depend on business metrics and business strategies, plus operational conditions. Both RESTful business process principles and MiniZnMASC increase software runtime adaptability. This algorithm also uses MiniZinc models and solver, but these models differ from the algorithm C because of different circumstances in which the adaptation decision is made.

III. DEMONSTRATION OF MINIZNMASC

Our demonstration shows how MiniZnMASC decides adaptation actions with different algorithms using a family of examples related to loan application. A lending institution provides a loan application process shown in Fig. 1. The institution classifies its consumers into three classes according to their previous loan records and credit history in the company: gold, silver, and bronze. The company provides different classes of consumer with different technical quality of service (QoS) guarantees, prices per year, and penalties if the guarantees are not met. There are different historical probabilities to meet these guarantees. There are also limits on the total cost of adaptations to be performed and on available IT resources to run business process instances.

In the examples for the algorithms A, B, and C, the credit check service is a 3rd-party service provided by an external credit check agency. During runtime, it suddenly becomes unavailable for some reason. The adaptation system finds two alternatives for the credit check service, CreditCheck1 and CreditCheck2. Each alternative provides 2 different classes of service (business and economy class), with various guaranteed QoS, prices, penalties, and historical probabilities to meet the guarantees. Setup times for these services also differ. There are also related BMM models capturing longer-term business considerations. All this information is captured in appropriate WS-Policy4MASC policy assertions. The algorithm B uses the BMM models, while the algorithm A does not, resulting in different adaptation decisions. For the algorithm C, classes of instance are determined by their current position in the business process and their class of consumer. There are 9 classes of instance (listed in [3]) and each has several running instances. The demonstration shows that the adaptation decisions made by the algorithm C have higher overall business value than the decisions made by the algorithm A sequentially applied 9 times in the same circumstances. This is because the algorithm A finds a set of solutions that are locally optimal each time the algorithm is run, but which due to global constraints are not globally optimal.

In the example for the algorithm D, there is a decision making point in the credit check service, which means 1 adaptation option should be chosen as the “next steps” process fragment. The lending institution provides three different process fragment options to deal with the credit check: use the external credit check process, use the internal credit check process, and skip the credit check process. In different business and technical circumstances, different process fragments are appropriate. In particular, different adaptation options are selected for different classes of user.

Our software demonstration first shows parsing of WS-Policy4MASC policies and events into the MiniZnMASC database. Then MiniZnMASC is run 4 times and decision results produced by the different decision making algorithms discussed in Section II are shown. The differences in the algorithms (causing different results) and the effort needed to substitute the algorithms in MiniZnMASC are discussed.
REFERENCES


