Compensation-Aware Runtime Monitoring*

Christian Colombo$^1$, Gordon J. Pace$^1$, and Patrick Abela$^2$

$^1$ Dept. of Computer Science, University of Malta, Malta
$^2$ Ixaris Ltd, Malta
{christian.colombo,gordon.pace}@um.edu.mt, patrick.abela@ixaris.com

Abstract. To avoid large overheads induced by runtime monitoring, the use of asynchronous log-based monitoring is sometimes adopted — even though this implies that the system may proceed further despite having reached an anomalous state. Any actions performed by the system after the error occurring are undesirable, since for instance, an unchecked malicious user may perform unauthorized actions. Since stopping such actions is not feasible, in this paper we investigate the use of compensations to enable the undoing of actions, thus enriching asynchronous monitoring with the ability to restore the system to the original state in which the anomaly occurred. Furthermore, we show how allowing the monitor to adaptively synchronise and desynchronise with the system is also possible and report on the use of the approach on an industrial case study of a financial transaction system.

1 Introduction

The need for correctness of systems has driven research in different validation and verification techniques. One of the more attractive approaches is the use of monitors on systems to verify their correctness at runtime. The main advantage in the use of runtime verification over other approaches, is that it is a relatively lightweight approach and scales up to large systems — guaranteeing the observation of abnormal behaviour.

Even though monitoring of properties is usually computationally cheap when compared to the actual computation taking place, the monitors induce an additional overhead, which is not always desirable in real-time, reactive systems. In transaction processing systems, the additional overhead induced by each transaction can limit throughput and can cripple the user-experience at peak times of execution. One approach usually adopted in such circumstances, is that of evaluating the monitors asynchronously with the system, possibly on a separate address space. The overhead is reduced to the cost of logging events of the system, which will be processed by the monitors. However, by the time the monitor has identified a problem, the system may have proceeded further.

The problem is closely related to one found in long-lived transactions [14] — transactions which may last for too long a period to allow for locking of resources, but which could lead to an inconsistent internal state if the resources are released. To solve the

* The research work disclosed in this publication is partially funded by the Malta National Research and Innovation (R&I) Programme 2008 project number 052.

© Springer-Verlag Berlin Heidelberg 2010
problem, typically one defines *compensations*, to undo partially executed transactions if discovered to be infeasible half way through. In the case of asynchronous monitoring, allowing the system to proceed before the monitor has completed its checks may lead to situations where the system should have been terminated earlier. As with long-lived transactions, we allow this run-ahead computation. We adopt the use of compensations in our setting to enable the undoing of system behaviour when an asynchronous monitor discovers a problem late, thus enabling the system to rollback to a sane state. Furthermore, in a setting such as transaction-processing systems, one can afford most of the time to run the monitors in synchrony with the system, falling back to asynchrony only when required due to high system load. Thus, we propose an architecture to enable loosely-coupled execution of monitors with the system, typically running synchronously, but allowing for de-synchronisation when required and re-synchronisation when desired.

In this paper, we present a framework to enable compensation-aware monitoring — and prove that the compensation triggering mechanism works as expected, resulting in similar behaviour as though we had run the monitor synchronously. Furthermore, we show that enabling the monitor to synchronise (and de-synchronise) at will with the system does not change the behaviour. We have investigated the use of this approach on an industrial case study — dealing with financial transactions, and for which a compensation-based implementation was already in place.

The paper is organised as follows — in section 2 we present background necessary to reason about compensations, which we use to formally characterise compensation-aware monitoring in section 3. An architecture implementing this mode of monitoring is presented in section 4 and we illustrate its use on an industrial case study in section 5. Finally we discuss related work in section 6.

## 2 Compensations

Two major changes occurred which rendered traditional databases inadequate in certain circumstances [14][13]: on the one hand there was the advent of the Internet, facilitating the participation of heterogeneous systems in a single transaction, and on the other hand, transactions became longer in terms of duration (frequently, the latter being a consequence of the former). These changes meant that it was possible for a travel agency to automatically book a flight and a hotel on behalf of a customer without any human intervention — a process which may take time (mainly due to communication with third parties and payment confirmation) and which may fail. These issues rendered the traditional mechanism of resource locking for the whole duration of the transaction impractical since it may cause severe availability problems, and motivated the need for a more flexible way of handling transactions amongst heterogeneous systems while at the same time ensuring correctness. A possible solution is the use of compensations [14][13] which are able to deal with partially committed long-lived transactions with relative ease. Taking again the example of the flight and hotel booking, if the customer payment fails, the agency might need to reverse the bookings. This can be done by first cancelling the hotel reservation followed by the flight cancellation, giving the impression that the bookings never occurred. Although several notations supporting compensations have