ABSTRACT

This research has defined an abstract execution model for establishing user-defined correctness and recovery in a service composition environment. The service composition model defines a flexible, hierarchical service composition structure, where a service is composed of atomic and/or composite groups. The model provides multi-level protection against service execution failure by using compensation and contingency at different composition granularity levels, thus maximizing the potential for forward recovery of a process when failure occurs. The recovery procedures also include rollback as a recovery option, where incremental data changes known as deltas are extracted from service executions and externalized by streaming data changes to a Process History Capture System. Deltas can then be used to backward recover an operation through a process known as Delta-Enabled Rollback. This article defines the semantics of the service composition model and the manner in which compensation, contingency, and Delta-Enabled-rollback are used together to recover process execution. The authors also present a case study and describe a simulation and evaluation framework for demonstrating the functionality of the recovery algorithm and for evaluating the performance of the recovery command generation process. [Article copies are available for purchase from InfoSci-on-Demand.com]

Keywords: Compensation; Contingency; Delta-Enabled Rollback; Failure Recovery; Service Composition

INTRODUCTION

In a service-based architecture, a process is composed of a series of calls to distributed Web services and Grid services that collectively provide some specific functionality of interest to an application (Singh & Huhns, 2005). In a traditional, data-oriented, distributed computing environment, a distributed transaction is used to provide certain correctness guarantees about the execution of a transaction over distributed data sources. In particular, a traditional, distributed transaction provides all-or-nothing behavior by using the two-phase commit protocol to support...
atomicity, consistency, isolation, and durability (ACID) properties (Kifer et al., 2006). A process in a service-oriented architecture, however, is not a traditional ACID transaction due to the loosely-coupled, autonomous, and heterogeneous nature of the execution environment. When a process invokes a service, the service performs its function and then terminates, without regard for the successful termination of the global process that invoked the service. If the process fails, reliable techniques are needed to either 1) restore the process to a consistent state or 2) correct critical data values and continue running.

Techniques such as compensation and contingency have been used as a form of recovery in past work with transactional workflows (e.g., Worah & Sheth, 1997) and have also been introduced into recent languages for service composition (e.g., Lin & Chang, 2005). In the absence of a global log file, compensation provides a form of backward recovery, executing a procedure that will “logically undo” the affects of completed and/or partially executed operations. Contingency is a form of forward recovery, providing an alternate execution path that will allow a process to continue execution. Some form of compensation may be needed, however, before the execution of contingency plans. Furthermore, nested service composition specifications can complicate the use of compensating and contingent procedures. To provide a reliable service composition mechanism, it is important to fully understand the semantics and complementary usage of compensation and contingency, as well as how they can be used together with local and global database recovery techniques and nested service composition specifications.

This research has defined an abstract execution model for establishing user-defined correctness and recovery in a service composition environment. The research has been conducted in the context of the DeltaGrid project, which focuses on building a semantically-robust execution environment for processes that execute over Grid Services (Xiao, 2006; Xiao, Urban, & Dietrich, 2006; Xiao, Urban, & Liao, 2006; Xiao & Urban, 2007; Xiao & Urban, 2008). This article is an extended version of the work presented in (Xiao, Urban, and Liao, 2006), with a focus on the full specification of the abstract service composition and recovery model for the DeltaGrid environment.

The service composition model defines a flexible, hierarchical service composition structure, where a service is composed of atomic and/or composite groups. An atomic group is a service execution with optional compensation and contingency procedures. A composite group is composed of two or more atomic and/or composite groups and can also have optional compensation and contingency procedures. A unique aspect of the model is the provision of multi-level protection against service execution failure by using compensation and contingency at different composition granularity levels, thus maximizing the potential for forward recovery of a process when failure occurs.

Another unique aspect of the model is the support it provides for rollback as a recovery option. Distributed services in the DeltaGrid environment, referred to as Delta-Enabled Grid Services, are extended with the capability of recording incremental data changes, known as deltas (Blake, 2005; Urban et al., 2009). Deltas are extracted from service executions and externalized by streaming data changes out of the database to a Process History Capture System (PHCS) (Xiao, Urban, & Dietrich, 2006). The PHCS merges deltas from distributed sources into a time-ordered schedule of the data changes associated with concurrently executing processes. Deltas can then be used to backward recover an operation through a process known as Delta-Enabled Rollback (DE-Rollback) (Xiao, 2006). DE-rollback can only be used, however, if certain recoverability conditions are satisfied, with the PHCS and the merged schedule of deltas providing the basis for determining the applicability of DE-rollback based on data dependencies among concurrently executing processes.

Our work defines the semantics of the service composition model and the manner in which compensation, contingency, and DE-
Service Oriented Solution Modeling and Variation Propagation Analysis Based on Architectural Building Blocks
www.igi-global.com/article/service-oriented-solution-modeling-and-variation-propagation-analysis-based-on-architectural-building-blocks/108881?camid=4v1a

Behavioral Attestation for Web Services Based Business Processes
www.igi-global.com/article/behavioral-attestation-web-services-based/45176?camid=4v1a