WEB SERVICE TRANSACTION CORRECTNESS

Aspen Olmsted

In this research we investigate the problem of providing consistency, availability and durability for Web Service transactions. First, we show that the popular lazy replica update propagation method is vulnerable to loss of transactional updates in the presence of hardware failures. We propose an extension to the lazy update propagation approach to reduce the risk of data loss. Our approach is based on the ”buddy” system, requiring that updates are preserved synchronously in two replicas, called buddies. The rest of the replicas are updated using lazy update propagation protocols. Our method provides a balance between durability (i.e., effects of the transaction are preserved even if the server, executing the transaction, crashes before the update can be propagated to the other replicas) and efficiency (i.e., our approach requires a synchronous update between two replicas only, adding a minimal overhead to the lazy replication protocol). Moreover, we show that our method of selecting the buddies ensures correct execution and can be easily extended to balance workload, and reduce latency observable by the client.

Second, we consider Web Service transactions that consume anonymous and attribute based resources. We show that the availability of the popular lazy replica update propagation method can be achieved while increasing its durability and consistency. Our system provides a new consistency constraint, Capacity Constraint, which allows the system to guarantee that resources are not over consumed and also allows for higher distribution of the consumption. Our method provides; 1.) increased availability through the distribution of element master’s by using all available clusters, 2.) consistency by performing the complete transaction on a single set of
clusters, and 3.) guaranteed durability by updating two clusters synchronously with the transaction.

Third, we consider each transaction as a black box. We model the corresponding metadata, i.e., transaction semantics, as UML specifications. We refer to these WS-transactions as coarse grained WS-transactions. We propose an approach that guarantees the availability of the popular lazy replica update propagation method while increasing the durability and consistency. In this section we extend the Buddy System to handle course grained WS-transactions, using UML stereotypes that allow scheduling semantics to be embedded into the design model. This design model is then exported and consumed by a service dispatcher to provide: 1.) High availability by distributing service requests across all available clusters. 2.) Consistency by performing the complete transaction on a single set of clusters. 3.) Durability by updating two clusters synchronously.

Finally, we consider enforcement of integrity constraints in a way that increases availability while guaranteeing the correctness specified in the constraint. We organize these integrity constraints into three categories: entity, domain and hierarchical constraints. Hierarchical constraints offer an opportunity for optimization because of an expensive aggregation calculation required in the enforcement of the constraint. We propose an approach that guarantees that the constraint cannot be violated but it also allows the distribution of write operations among many clusters to increase availability. In our previous work, we proposed a replica update propagation method, called the Buddy System, which guaranteed durability and increased availability of web services. In this section we extend the Buddy System to enforce the hierarchical data integrity constraints.