Dynamic Restructuring of Transactions

Gail E. Kaiser and *Calton Pu*
Columbia University
Department of Computer Science
500 West 120th Street
New York, NY 10027
kaiser@cs.columbia.edu, 212-854-3856
calton@cs.columbia.edu, 212-854-8110
Fax: 212-666-0140

CUCS-012-91
April 1991
(revised August 1991)

Abstract

Open-ended activities are characterized by uncertain duration, unpredictable developments, and interactions with other concurrent activities. Like other database applications, they require consistent concurrent access and fault-tolerance, but their unconventional characteristics are incompatible with the conventional database mechanisms of concurrency and failure atomicity. We present the split-transaction and join-transaction operations for *restructuring* in-progress transactions, as an approach to consistent concurrent access and fault-tolerance for open-ended activities. Split-transaction divides an on-going transaction into two or more transactions that are serializable with respect to each other and all other transactions, and each of the new transactions is later committed or aborted independently of the others. Join-transaction merges two or more transactions that are serializable with respect to each other into a single transaction as if they had always been part of the same transaction, and all their work is now committed or aborted together. Split-transaction is useful for committing some work early or dividing on-going work among several co-workers. Join-transaction allows to hand over results to a co-worker to integrate into his or her own ongoing task. The transaction manager enforces that the new transaction(s) will in fact be serializable, and does not permit the split or join otherwise.

To appear in Ahmed K. Elmagarmid, ed., *Database Transaction Models for Advanced Applications*, chapter 8, Morgan Kaufmann, 1991.