KOHLER81

W.H. Kohler, "A Survey of Techniques for Synchronization and Recovery in Decentralized Computer Systems", Computing Surveys, June 1981, Volume 13, No. 2, pp. 149-183 This paper is a survey of techniques that have been proposed to address two fundamental problems in the design of reliable decentralized computer systems. The first problem is synchronizing access to shared objects while supporting a high degree of concurrency. The second is the recovery of objects in spite of user, application or partial system errors. When talking of synchronization, it is usually referred to the solution of either the specification and control of cooperating sequential processes or to the serialization of concurrent access to shared objects by multiple processors. Proposed solutions to either problems frequently address both problems, but in general conventional tools like semaphores, critical sections and monitors represent solution to the specification and control problem. This paper concentrates on the solution of the serialization of access to objects by multiple processes, also referred to as concurrency control. An object is an element of computation whose state can only be modified by a pre-determined set of operations, that in database terminology are called transactions. Synchronization techniques are meant to provide a way of interleaving non-conflicting steps so that transactions can proceed with maximum concurrency. In general we are interested in a consistent schedule, i.e. a sequence of operations such that the effect of running a sequence of concurrent transactions is the same as if the transactions had been run serially. In this paper synchronization techniques have been divided into four categories according to their most distinguishing feature: locking, timestamps, circulating permits and tickets and conflict analysis. With a locking scheme, a transaction may lock objects while in a temporarily inconsistent state. If a transaction attempts to lock an object that is already locked, it must either wait, abort or pre-empt another transaction. When two or more transactions are waiting, deadlock may occur. In distributed systems there may be local lock managers and centralized locking controllers and each must cooperate to detect global deadlock. Deadlock may be broken by preemption or by time-outs to revoke waits after a specified time interval. An important issue is also the choice of locable objects. There may be different types of locks, too. For the purpose of reading-only, a share mode may be assumed, while objects locked in exclusive mode can be read and modified. The locking approach seems to be the most promising for extensions. A timestamp is a unique number assigned to a transaction and is chosen from an always increasing sequence. It is often used in conjunction with locks. The choice of a particular action is based on a comparison of the time stamps of the requesting and conflicting transactions. In the approach based on circulating permits and tickets, the nodes of a distributed system are linked in a ring on which a unique control token circulates. Only the node owning the token is allowed to perform a transaction. Tickets allow a higher degree of concurrency. The conflict analysis methodology, developed by the Computer Corporation of America, consists of a collection of database sites interconnected through a communication network. It is believed that this approach allows more concurrency than the classical locking approach. The concept of atomic action plays an important role in the problem of designing techniques for effort detection and recovery. An atomic action may be defined as a computation that may not be decomposed, i.e. there will not be observable intermediate state changes outside of the atomic action. If a failure occurs which prevents the successful completion of an atomic action must be restored to their initial state. Under this constraint, updated objects are not released until the action is completed and the initial states can be reconstructed. Two methods are described: the first is a model to quantify the notion of reliability of a system; the second, is an implementation of atomic action recoveries. The reliability of a system is computed by means of a modelling approach. The procedure is tedious and error prone, but it seems to be the only valid methodology available now. The atomic action recovery implementation is based on work done by Gray at the IBM Research laboratory in San Jose' Calif. This paper presents a fairly large number of studies and proposals, some of them in good details. It is a good review for people with a basic knowledge of concurrency formulations that want to expand their ideas on concurrency to the new challenge of distributed systems.