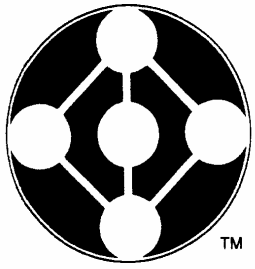

Alternative Technologies

Enterprise Integrity: Business Transactions VII

Vol. 6, No. 9

Over the last six months I've covered the fundamentals necessary for an understanding of physical, logical, and business transactions, then integrated that understanding through new, generalized ACID properties, and discussed certain aspects of current business transaction standards. This month, we'll finish the series by introducing some of the concepts behind collaborative transactions, how those concepts address specific business transaction issues such as problems regarding standards like the Business Transaction Protocol (BTP).

As noted earlier in this series, business transactions often require considerable flexibility regarding the conditions of success, resources used, participants, and even objectives. As with traditional distributed transactions, they may also require the participation of otherwise independent entities and resources. Unlike more familiar transactions, they are rarely well-defined in advance, but often evolve in the course of execution under an implicit collaborative agreement among the final participants. On the one hand, each participant in a business transaction usually desires a high degree of atomicity, integrity, consistency, and auditability after the fact. On the other hand, it may be impossible to demand any of these characteristics before the fact, simply because no one can define the transaction sufficiently well until it is, by mutual agreement, completed.

Clearly, business transactions require adaptive transaction management. By this I mean that transaction management must enforce transaction characteristics as they evolve, enable a high degree of controlled collaboration rather than blind isolation adherence, and offer error management techniques beyond predetermined actions such as rollback or compensation (e.g., corrective transactions). Actually, these properties are valuable for all types of transactions, even if they do not involve multiple business entities, multiple distributed resource managers, or redefinition. I call such transactions *collaborative transactions*. While it isn't possible to explain all the technological underpinnings of collaborative transactions here, I do want to describe two key concepts.

Collaborative transactions are enabled by the generalized ACID properties presented in my July column. Allowing us to understand business transactions as generalizations of logical transactions (like database transactions), they also allow us to define ways for transactions to share work and resources (that is, to collaborate) without suffering the ill effects of interference. It's easy to motivate this idea: Intermediate results often need to be shared with

collaborators without waiting for completion or failure our own tasks. However, the exchange must occur so that (a) the work is consistent in a mutually agreed upon manner, (b) any resources the work depends on are never under the control of more than one resource manager at a time, and (c) the collaborators cooperate in a safe recovery protocol. The main trick? Transfer work among collaborating transactions only at *consistency points*.

A consistency point is a potential commit point in the transaction with respect to the integrity of the work in question. Essentially, it is a savepoint with explicitly recorded consistency, whether manually or automatically identified. Consistency points enable efficient, automatic recovery from various in-flight transaction errors (e.g., deadlock) and are useful in scheduling transaction concurrency and intra-transaction parallelism. Most important, however, they enable collaborative transactions. At the consistency point, immediate responsibility for the relevant resources is transferred to the receiving transaction via a mechanism I call *transaction relaying*. Barring detailed explanation, suffice it to note that a transaction could have been designed to commit the relayed work, which could have then been read by the collaborating transactions. The logical properties of this sequence and that produced by transaction relaying are identical, and we can control whether or not work is shared among collaborating transactions. Careful analysis confirms that expensive two-phase commit among collaborating transactions isn't required, and the rollback path is well-defined.

Recognizing that even database consistency evolves and is sometimes context dependent, collaborative transactions also support negotiated consistency as long as the generalized ACID properties are satisfied. The key insight is that business transactions often try to achieve an acceptable result rather than a specific result, and that a group of transactions with a common initial state may have alternative final states determined by slight changes in the consistency conditions. Sets of consistency conditions that result in an acceptable termination form an equivalence class and thus a formal approach to negotiated consistency.

Collaborative transactions add the missing elements to a protocol like BTP. In particular, they prescribe the missing transaction mechanics necessary for safe use of such protocols in a consistent theoretical framework. Not only are they a way of formalizing the flexibility and adaptability necessary for real-world business transactions, but mange to preserve *enterprise integrity* in the process.

David McGoveran
Felton, CA