Transactional CSP Processes Demo

Gail Cassar[1] and Patrick Abela[1]

[1] Ixaris Systems (Malta) Ltd, Casa Roma, Sir Augustus Bartolo Street, Ta' Xbiex, Malta
{gail.cassar, patrick.abela}@ixaris.com

Abstract. *Long-lived transactions (LLTs)* are transactions intended to be executed over an extended period of time ranging from seconds to days. A long-lived transaction is normally organized as a series of *activities,* with each activity being a discrete transactional unit of work that releases transactional locks upon its execution. The long-lived transaction commits if all its activities complete successfully. Unless an activity requires the result of a previously committed activity, there is no constraint which specifies that the various activities belonging to a long lived transaction should execute sequentially.

We will present a framework called Transactional CSP Processes which provides a solution, implemented in Java, that combines long-lived transactions and CSP such that independent activities execute in parallel. This would achieve flexibility and better performance for long-lived transactions. The framework makes use of two composition constructs SEQ_LLT and PAR_LLT. Very much as the occam CSP-based constructs, SEQ and PAR, allow processes to be executed sequentially or concurrently, the proposed SEQ_LLT and PAR_LLT constructs can be used to specify the sequential or concurrent execution of transactions. Two activities that are coordinated with the SEQ_LLT construct are evaluated in such a way that the second activity is executed only after the first activity commits. This corresponds to the SEQ construct which, from a concurrency perspective, executes in such a way that the second process starts its execution after the first process is complete. Similarly, PAR_LLT specifies that activities can start their execution, independently from whether any other activities have committed their transaction or not.

Transactional CSP Processes provides an API through which the application developer can define long-lived transactions. Concurrency and transaction handling are managed by the framework transparently from the application developer. Therefore, we will be demonstrating how a developer would use the framework to define long lived transactions in terms of their activities and composition structure.

Keywords: CSP, transaction processing, parallel transactions, long-lived transactions, compensating actions.