Relaxed Transaction Processing

Munindar P. Singh, Christine Tomlinson, and Darrell Woelk
The Carnot Project
Microelectronics and Computer Technology Corporation
3500 W. Balcones Center Drive
Austin, TX 78759, USA
{msingh,tomlic,woelk}@mcc.com

1 Motivation

Extended transaction models have drawn much interest recently in academia and industry [2]. Such models seek to address the limitations of traditional ACID transactions for supporting multisystem applications that operate in heterogeneous environments. Such applications are increasingly proving to be of strategic importance to a number of businesses and governmental agencies. Different transaction models, however, tend to be closed in that they cannot be easily combined with other such models, thus limiting their applicability to situations which exactly match one of them.

We do not propose yet another transaction model. Instead, we have developed a general specification facility that enables the formalization of any transaction model that can be stated in terms of dependencies amongst significant events in different subtransactions. Such significant events include *start*, *commit*, and *abort*. We make no assumptions that these are the only kinds of events. Our approach is viable because most extended transaction models can be naturally formalized in terms of dependencies among different subtransactions.

2 Our Approach

Our approach begins with specifications stated in an extension of the language proposed recently by Klein [3]. These specifications are translated into an internal language for processing. The processing proceeds by provably sound algebraic manipulations and yields guards on the different significant events of interest. Our execution environment is Carnot's extensible services switch (ESS) [5]. The ESS, which is an implementation of the Actor model, supports point-to-point message transmission as well as multicasts over a heterogeneous environment. Therefore, it is an ideal medium over which to integrate diverse applications.

We associate an actor with each subtransaction or legacy application that mediates between that subtransaction and the rest of the system. The guards on events are located in these mediating actors. A subtransaction communicates with its associated actor to request permission to execute some controllable event, e.g., *commit*, and to inform the system of some uncontrollable event, e.g., *abort*. Additionally, the actor may trigger events, e.g., *start*, in the subtransaction.

Further, while flexibility was an important motivation for it, our approach is meant also to lead to efficient execution. Indeed, the present approach, which we will demonstrate, is not only more general than our previous approach [1], but also more efficient. There is almost always a certain performance penalty to pay for flexibility. We believe that this penalty is not too high for our present approach.

General descriptions of our approach, and details about the Carnot architecture and the ESS, are available in the literature, e.g., [4, 5].

References

[1] P. C. Attie, M. P. Singh, A. P. Sheth, and M. Rusinkiewicz. Specifying and enforcing intertask dependencies. In *19th VLDB*, August 1993.

[2] A. K. Elmagarmid, editor. *Database Transaction Models for Advanced Applications.* Morgan Kaufmann, 1992.

[3] J. Klein. Advanced rule driven transaction management. In *IEEE COMPCON*, 1991.

[4] C. Tomlinson, P. Attie, P. Cannata, G. Meredith, A. Sheth, M. Singh, and D. Woelk. Workflow support in Carnot. *IEEE Data Engineering*, June 1993.

[5] C. Tomlinson, G. Lavender, G. Meredith, D. Woelk, and P. Cannata. The Carnot extensible services switch (ESS) - support for service execution. MCC Technical Report Carnot-129-92, June 1992.