A Transaction Manager Component for Cooperative Transaction Models

Thesis Proposal

George T. Heineman

Columbia University
Department of Computer Science
500 West 120th Street
New York, NY 10027
heineman@cs.columbia.edu
TR CUCS-017-93

July 27, 1993

Abstract

The database community has produced extensive research on the concurrency control problem in the context of traditional databases. However, this traditional model is not suitable for some applications, such as CAD/CAM systems and software development environments. What is needed is an extended transaction model better suited for these newer applications. Unfortunately, there is no consensus as to which of the almost dozens of extended transaction models is appropriate. This proposal posits that there is no single model which will be applicable to every application. To solve this problem, we propose PERN, a flexible and tailorable transaction manager component which provides the necessary support required by advanced database applications. In addition we introduce a transaction language for defining transaction models.

©1993 George T. Heineman
# Contents

1 Introduction
   1.1 Introduction to Transactions ........................................... 2
   1.2 Motivation ........................................................................... 5
   1.3 Related Work ...................................................................... 7

2 Problems
   2.1 Scope ................................................................................. 10

3 Proposal
   3.1 Session Layer ...................................................................... 12
   3.2 Task Layer .......................................................................... 14
   3.3 PERN Layer ......................................................................... 16
      3.3.1 PERN’s recovery model .................................................. 18
   3.4 CORD rule language .......................................................... 19
   3.5 Multiversion Concurrency Control ........................................ 21
   3.6 Formal Model ...................................................................... 22
      3.6.1 PERN Transaction Model ................................................ 24

4 Feasibility
   4.1 Implementing SCCP with coordination rules ............................ 26
      4.1.1 Evaluation of Coordination Rule Approach .......................... 28
   4.2 Compiling a set of coordination rules ..................................... 28
   4.3 Integrating PERN .................................................................. 29
   4.4 Constructing PERN’s formal model ....................................... 30
   4.5 Generality of PERN’s formal model ....................................... 31

5 Marvel
   5.1 Decomposition of conflict detection and resolution ..................... 34
   5.2 Abstraction of Semantic Correctness ....................................... 36
   5.3 Nested Transaction model applied to Rule Chaining ..................... 37
   5.4 Limitations of Consistency Model ......................................... 37
   5.5 Limitations of Consistency Model ......................................... 39

6 Contribution ........................................................................... 41

A SCCP environment ..................................................................... 42

B Compiled CORD rules .................................................................. 43

C Interface Specification
   C.1 Session Layer services ......................................................... 46
   C.2 SP Interface ......................................................................... 47
   C.3 Task Layer services ............................................................. 48
List of Figures

1  Sample Cooperative Process ................................................. 5
2  Motivation for Transaction Manager Component .......................... 7
3  Organization of GENESIS prototype ........................................ 8
4  Organization of Mneme system .............................................. 8
5  “Toaster” Reference Model .................................................. 10
6  PERN four-layered Architecture ............................................. 13
7  Session Layer Algorithm ..................................................... 15
8  TL Algorithm ................................................................. 15
9  PL Algorithms ................................................................. 17
10 CRL syntax ................................................................. 19
11 Possible Split Points .......................................................... 20
12 Four Case Studies ............................................................ 29
13 Task Layer specification of sagas ............................................ 32
14 Rule Chaining and Nested Transactions ................................... 38
15 higher_or_same_priority ..................................................... 44
16 higher_priority .............................................................. 44
17 Interface Specification ....................................................... 46
Introduction

1 Introduction

This thesis proposal describes a transaction manager component, called PERN, with a tailorable concurrency control protocol. PERN will provide a framework for integrating extended transaction models with conventional systems. A formal model is introduced, capturing many of the features of previous extended transaction models, and the CORD language is described, which allows for the programming of a particular concurrency control policy. PERN’s ultimate aim is to provide support for cooperation by allowing an administrator to define correct concurrent behavior of cooperating agents.

The database community has produced extensive research on the concurrency control problem in the context of traditional databases. The two central concepts developed to solve this problem are Transactions and Serializability [18, 10]. When multiple transactions execute concurrently in the database, their operations are interleaved, producing a Schedule. Through these concepts, the concurrency control problem reduces to the testing of whether a schedule of transactions is serializable. Some parts of the database community are essentially satisfied with the traditional transaction model, as they focus their research on other advanced database functionality [46].

There is, however, expressed concern [33, 12, 8] that this traditional model is not suitable for advanced applications, such as CAD/CAM systems and software development environments (SDEs). These applications typically involve long interactive database sessions and cooperation among multiple users. Open-ended activities, such as editing and designing, are unnecessarily restricted by the traditional transaction model. Cooperation is essentially denied since interaction among ongoing transactions is explicitly prohibited by the concurrency atomicity property of traditional transactions.

What is needed is an extended transaction model better suited for these newer applications. Unfortunately, there is no consensus as to which of the almost dozens of extended transaction models is appropriate. This proposal posits that there is no single model which will be applicable to every application. Therefore, instead of defining yet another transaction model, PERN is a transaction manager component capable of supporting a wide variety of extended transaction models.

This thesis proposal is organized as follows. We discuss the traditional model of transaction processing, introducing the concepts of transactions and serializability. We motivate this proposal with a brief example and a discussion of related research showing the need for such work. We produce an architecture of PERN and a corresponding transaction model supporting PERN. We conclude with a clear definition of the scope of this thesis and its contributions.