## ABSTRACT OF THE DISSERTATION

Application-Transparent Error Recovery Techniques for Multicomputers

by

Tiffany Michelle Frazier
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 1995
Professor Yuval Tamir, Chair

This dissertation addresses the problem of providing application-transparent error recovery to restore a valid system state in general-purpose scalable multicomputers following hardware failures. Message logging and coordinated checkpointing are the two main classes of error recovery techniques that provide transparent checkpointing and rollback of distributed applications. Several new algorithms are presented, their correctness proven, and their performance evaluated.

A new message logging algorithm uses *volatile message logs* to recover as much of the failed process' pre-failure execution as possible. Checkpointing of process states and logging of messages proceeds asynchronously with respect to process execution. This scheme is shown to be *correct* — in the presence of failures it can be used to recover one or more processes of a distributed application, achieving a *valid* system-wide application state.

Two new algorithms based on coordinated checkpointing of interacting sets of processes are presented and proven correct. These algorithms checkpoint multiple processes ''simultaneously'' such that their saved states are consistent. Logging of messages is not required. The first algorithm requires synchronous checkpointing —

while processes are checkpointed to stable storage, their execution is suspended. The second algorithm, *asynchronous coordinated checkpointing*, suspends process execution only during the taking of a *local volatile checkpoint*. Like many *independent checkpointing* techniques, the fraction of time a process is suspended is independent of network delays and the number of processes that constitute the application.

To evaluate the performance of these techniques we built DERT, a Distributed Error Recovery Testbed. DERT is built on top of an event-driven, multithreaded simulation kernel. Actual compiled distributed applications are executed on a simulated multicomputer. Instrumentation of the application assembly code is used to account for execution time and allow accurate simulation of periodic checkpointing. The testbed is fully functional — it can use one of several recovery schemes to restore one or more processes when simulated failures occur. Using this testbed we measured the impact of the recovery schemes and optimizations of these schemes on the error-free execution time of coarse-grained and fine-grained applications.