Implementation of Distributed Orthogonal Persistence Using Virtual Memory

Francis Vaughan

Thesis submitted for the degree of
Doctor of Philosophy
in
The University of Adelaide
(Faculty of Mathematical and Computer Sciences)

December 1994
Abstract

Persistent object systems greatly simplify programming tasks, since they hide the traditional distinction between short-term and long-term storage from the applications programmer. As a result, the programmer can operate at a level of abstraction in which short-term and long-term data are treated uniformly. In the past most persistent systems have been constructed above conventional operating systems and have not supported any form of distributed programming paradigm. In this thesis we explore the implementation of orthogonally persistent systems that make direct use of the attributes of paged virtual memory found in the majority of conventional computing platforms. These attributes are exploited to support object movement for persistent storage to addressable memory, to aid in garbage collection, to provide the illusion of larger storage spaces than the underlying architecture allows, and to provide distribution of the persistent system.

The thesis further explores the different models of distribution, notably a one world model in which a single persistent space exists, and a federated one in which many co-operating spaces exist. It explores communication mechanisms between federated spaces and the problems of maintaining consistency between separate persistent spaces in a manner which ensures both a reliable and resilient computational environment. In particular characterising the interdependencies using vector clocks and the manner in which vector time can be used to provide a complete mechanism for ensuring reliable and resilient computation.

The thesis concludes with a description of a new operating system design in which support for the mechanisms described earlier are intrinsic in the design. This operating system is able to provide orthogonal persistence in a distributed environment with no effort on the part of users of the operating system.
## Contents

### Chapter 1. Introduction.
- Persistence ............................................. 1
- Accidents of History .............................. 4
- Models of persistence ......................... 5
  - Specially designated objects ........... 5
  - Embedded persistence .................. 7
  - Reachability ..................................... 7
- Distribution ........................................ 8
  - Single Unstructured ......................... 8
  - Structured single address space .... 9
  - Fully partitioned ............................ 9
  - Federated ....................................... 10
  - Difficulties .................................... 10
- Implementation Mechanisms ................. 11
  - Specialised Hardware .................... 11
  - Software ....................................... 11
  - Conventional Hardware ................. 12
    - Granularity ............................... 12
    - Native code support ................ 13
- The Thesis ....................................... 13
  - Contributions ............................... 14

### Chapter 2. Implementation Tactics.
- Introduction ..................................... 17
- Issues ................................................ 17
  - Data Movement ................................. 18
    - Snapshots, Boot and Resume, and Undump. 18
    - Data Movement in Persistent Systems 19
- Address Generation ........................... 19
- Pointer Representation ....................... 20
- Garbage Collection .............................. 20
  - Generation GC ................................ 21
  - GC in Persistent Systems .......... 22
- Basic architectures ............................ 22
  - Object Tables ................................ 23
- Hardware .......................................... 24
- Monads ............................................. 25
- Lisp Machines .................................... 26
  - Data Format .................................... 26
  - Address Generation and Data Movement . 27
  - Garbage Collection ......................... 27
- ORSLA .............................................. 28
  - Data Formats .................................. 28
  - Garbage Collection .......................... 29
  - Data Movement ................................. 30
- Rekursiv .......................................... 30
  - Data Movement ................................ 31
  - Object Format ................................ 32
  - Garbage Collection ......................... 33
- Software .......................................... 33
  - Smalltalk ....................................... 34
  - Recognising Pointers .................... 35
  - LOOM .......................................... 36
  - Smalltalk and Mneme ...................... 38
  - Garbage Collection ......................... 41
- PS-algol and CPOMS ......................... 42
  - Computational model .................... 42
Chapter 5. Implementation Strategies.

5.1  Introduction  125
5.2  Related Work.  125
  5.2.1  Shrines  125
  5.2.2  Page Based Napier88 Stores  125
  5.2.3  Browns Before-Look Store  125
        5.2.3.1  Operation  126
  5.2.4  Munro's After-Look Store  127
  5.2.5  Thatte  129
  5.2.6  Logging Systems  131
  5.2.7  RVM  131
  5.2.8  Texas  132
  5.2.9  Meme  133

5.3  CASPER Bi-Phase  135
  5.3.1  Basics  135
  5.3.2  Page Map.  136
  5.3.3  Store Structure  137
  5.3.4  Store Creation  139
  5.3.5  Secondary Files  140
  5.3.6  Store Start-up  140
  5.3.7  Napier88 Implementation  141
  5.3.8  Runtime Actions  142
        5.3.8.1  Page States  142
        5.3.8.2  Read Access  142
        5.3.8.3  Write Access  144
        5.3.8.4  LPMAP mapping  145
  5.3.9  Meld  146
  5.3.10 Recovery  147
  5.3.11 Implementation Specifics  148
        5.3.11.1  Placement in memory  148
        5.3.11.2  Page copy sequence  148
        5.3.11.3  Memory Allocation and Swap Space.  149
        5.3.11.4  Implementation under Mach  149
        5.3.11.4.1  External Pager  150
        5.3.11.4.2  Problems with Mach  151
  5.3.12 Partial State Meld  152
  5.3.13 Multi-Phase  154
        5.3.13.1  Store Modifications  155
        5.3.13.2  Start-up  155
        5.3.13.3  Normal Running  156
        5.3.13.4  Meld Protocol  156
        5.3.13.5  Snapshot Deletion  157

5.4  Comparisons and Conclusions  158
  5.4.1  Log structured versus shadowing  158

Chapter 6. Distributed Casper.

6.1  Introduction.  161
6.2  Execution Environment.  161
  6.2.1  Stability and Coherency  162
6.3  Overview of the Architecture  162
6.4  Stable store server  164
  6.4.1  Store Stability  164
  6.4.2  Associations  165
  6.4.3  Stabilization  168
  6.4.4  Stable Store Heap Management  169
6.5  Clients  170
  6.5.1  External Pager  170
  6.5.2  Atomic Access  172
  6.5.3  Local Heap Management  173
6.6  Cache coherency  174