An Infrastructure for Multiprocessor Run-Time Adaptation

J. Appavoo, K. Hui, M. Stumm (Univ of Toronto)
R. Wisniewski, Dilma Da Silva, O. Krieger (IBM Watson)
C. Soules (CMU)

WOSS'02

self healing meaning in our project

- selectively upgrade components with bug, security, or performance fixes without down-time
- replace active component with the most appropriate implementation based on concurrent or expected use
- involves flexible/dynamic scheme for "self-diagnosis" (monitoring)

Part of self-healing problem we are dealing with

- system software approach; focus on operating systems (particularly demanding!)
- efficient, scalable, flexible mechanisms and infrastructure
- context: K42 research operating system
  - 64-bit cache-coherent shared memory multiprocessors
  - object-oriented component model for scalability to hundreds/thousands of processors
  - open source project; Linux API/ABI compatible

Applications we are targeting

- usual system benchmarks & scientific apps
- Hot-swapping for:
  - patches and updates
  - OS performance optimizations for common case
  - adaptive algorithms
  - application specific optimizations
  - third-party modules

What we achieved so far

- hot-swapping infrastructure is fully functional
- key virtual memory and file system components are hot-swapped based on runtime demands => significant performance gains
- key technologies (described in paper):
  - component model: clustered objects
  - hot-swapping algorithm (scalable, efficient)
  - method for detecting quiescence
    - K42 & Sequent NumaQ; adopted in Linux 2.5

We are not dealing with:

- generic state transfer
- safety issues in dowloading new components
- verification of "component equivalence"
- how to express tradeoffs between implementations in a generic way => automated choices (non ad-hoc)
- identifying language support benefits