Self-updating Software

Barbara Liskov & Daniel Jackson

Progress Report, June 1999

overview

motivation

- pervasive computing
  - personal devices, appliances, sensors & actuators
- dynamic reconfiguration
  - change in needs, location, resources
  - fixing errors and upgrading function

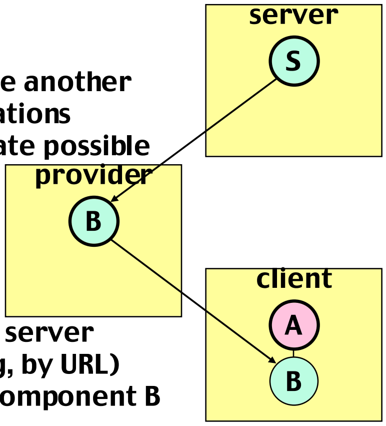
challenges

- safety & reliability
  - updates are consistent & reversible
- automation
  - minimal user effort
- performance
  - no service interruption
  - scales to large distributed systems
- multiple component suppliers

progress: dependency model

dependency model
- decouples components from one another
- components chosen by specifications
- makes automated repair & update possible

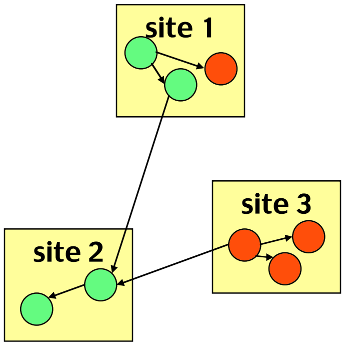
runtime scheme
- user requests new functionality
  eg, component A needs S
- client machine looks in local DB
- if not found, requests spec from server
  server identifies provider (eg, by URL)
- client downloads & integrates component B



progress: object update

context
- in context of Thor, an OODB
- huge state, widely distributed
  trillions of objects
  millions of clients
- update should be atomic

lazy update scheme
- update treated as a transaction
- object only updated when necessary
- global state may be inconsistent
  but user transactions never notice



circles are individual objects
arrows are object refs
green newer than red

plans (next 6 months)

dependency model
- finish design
- build initial infrastructure
- demonstrate on
  - Java packages
  - small shareware programs

object update
- refine design
- build scheme for lazy updates
  - good performance
  - clean & desirable semantics
- demonstrate on
  - Thor persistent object store