Dynamic Communities

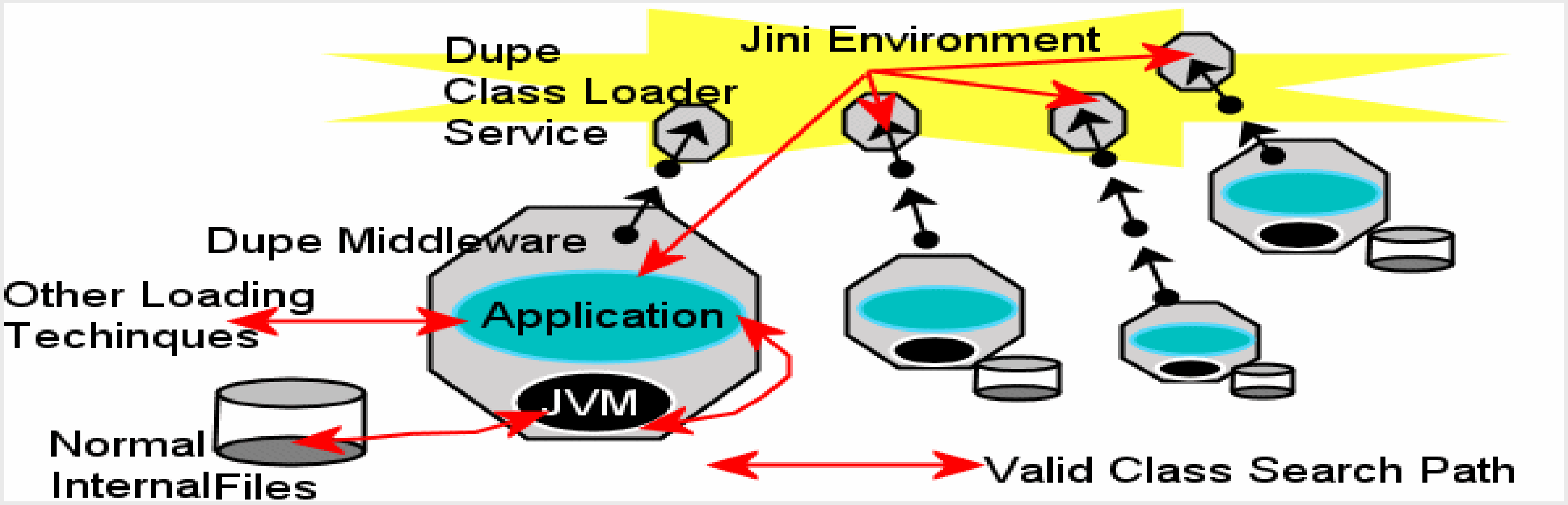
Using Dupe Middleware Techniques

Adrian Ryan

adrianr@infotech.monash.edu.au
http://phd.netcomp.monash.edu.au/adrian

Discovery based Remote Class Loading

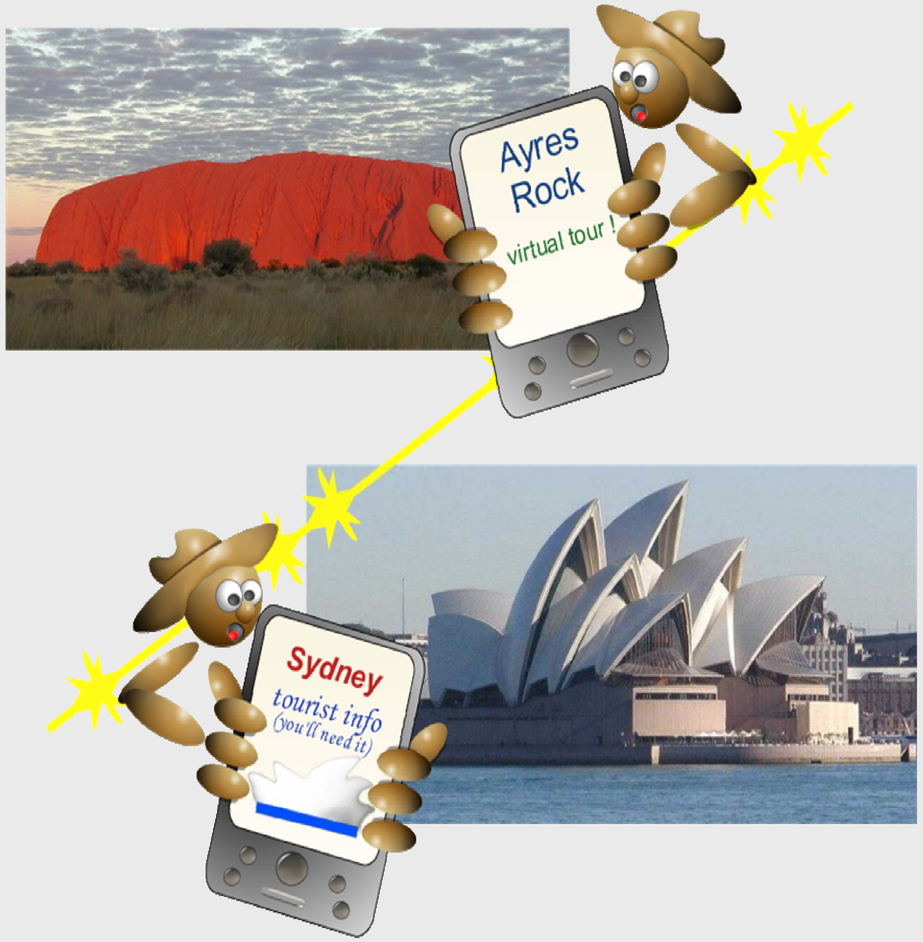
- Dupe applies Jini discovery mechanisms to extend an application's class loading reach beyond its own file system by advertising Proxies to controlling Remote Class Loaders
- Although restricted to a common mechanism/task, unlike agents, which suffer from a similar limitation, each application knows exactly what to say and do, *share code*



Using Remote Class Loading

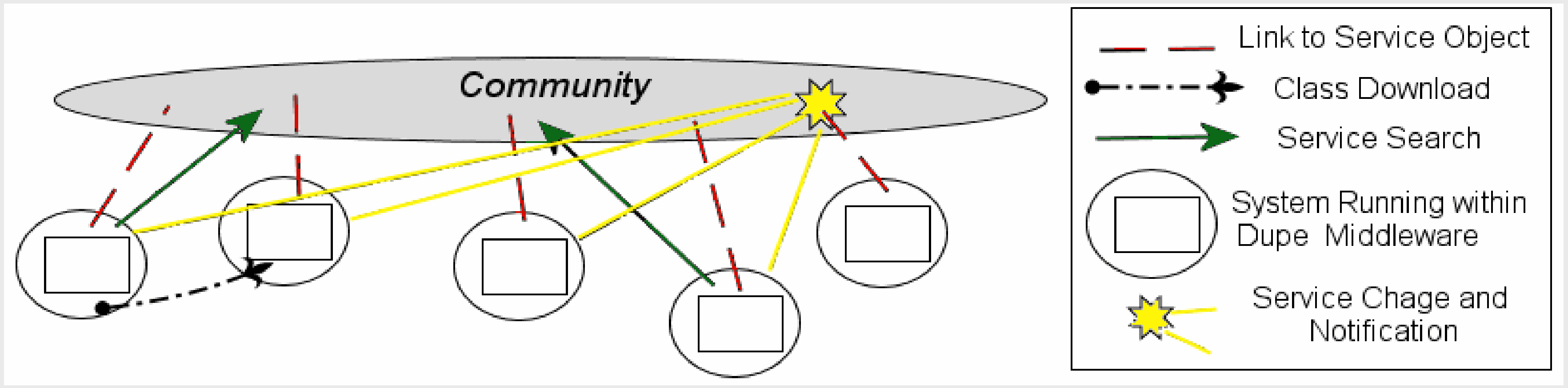
A Tourist Application Example

- A Tourist Application that acts according to encountered *Destinations*
- Each specific destination contains all classes required for their needs
- The single system may act completely differently depending on current tourist location
- Classes will be accessed via ether a *local tourist system* or another *interacting device*



Community Class Coordination

- There is a continual commonality of all Class Loader (Proxy) Jini Services
  - Giving access to all classes from compatible systems within environment
- All Advertising Systems are able to discover other advertised services
- Generating a community of systems where all <u>classes</u> within all cooperating *systems* are accessible to all cooperating *systems*


Community
Link to Service Object
Class Download
Service Search
System Running within Dupe Middleware
Service Chage and Notification

Changing Community

Major points which allow Community Dynamisms and Dynamic Adaptation when moving between Communities

- Event detailing the Discovery of New Class Details
  - Using Jini Event Notifications
- Ability to remotely check new class version* details prior to downloading
  - Using Jini Service Entry Details
- Aided by the ability to Dynamically reload an old class as a new version*, keeping instantiated object states
- Change in system algorithm in accordance with current community

*Version details are defined by associated Jar Manifest details. A system is able to check against all Specification and Implementation details

Dupe Compliancy Concept

- Any Compatible Dupe Middleware must discover and advertise* their class loading structure yet what each does locally may be individualistic
- A Middleware technique designed for different JVM’s allows them all to work together, discovery and use** all available class details
- Further Community Mechanisms may include advanced concepts such as, for example:
  - System Adaptation, Negotiation Control, Code Optimisers, Code Randomises, Community Specific Code Support, Context based Code Discovery, ???

* Limited Resource Devices May wish to Disable Code Upload due to their Physical Constraints

** Research into the use of class files written for different JVM’s needs to be explored

Versioning Dupe Middleware

Dynamic Class Alteration

- All systems within the Community should cooperate, yet all system may not be running on the same JVM.
- For Example, different techniques allow systems to alter their class details dynamically. Dupe allows such techniques to discovery and access new class details throughout the community
  - Dupe 5.0 - *Java 5.0 java.lang.instrument*
  - Dupe JPDA - *JPDA JVM Introspection*
  - Possibly the use of JDrums[andersson2000] and other Secondary Products

[andersson2000] J. Andersson and T.Ritrau, Dynamic Code Update in JDrums. *Fourth International Software Architecture Workshop (ISAW'4) in Conjunction with ICSE'2000*, Limerick, Ireland, June 2000

Implications of Dupe

- Security Control is Difficult in a Dupe Community
  - Code is anonymously shared therefore Trojans are possible
  - Tight security controls limit the usefulness of the Community
- Code Consistency
  - Loose versioning may lead to inconsistent code version within an application

Associated Research Concepts

- Giving Limited storage devices the ability to access Class Files from within the Community

- Combining Dupe Concepts and Context Awareness
  - Altering state using classes from within the Community due to Environment Context Interaction