Redesigning and Enhancing the 
**UWAgent** Execution Engine

Duncan Smith  
Dr. Munehiro Fukuda  
CSS 497 Autumn Colloquium  
Friday, December 9th, 2005
What is UWAagent

- Java-based mobile agent execution platform
- UW Bothell Distributed Systems Laboratory
- Supports the AgentTeamwork grid computing middleware system
Project Accomplishments

• Replaced Java RMI with Java sockets
• Implemented three new features
  – Navigation over gateways
  – Monitor commands
  – Secure Communication
• Tested for class name collision
• Refactored existing code
Why Use Mobile Agents

1. Reduce network load
2. Overcome network latency
3. Encapsulate protocols
4. Execute asynchronously and autonomously
5. Adapt dynamically
6. Naturally heterogeneous
7. Robust and fault-tolerant
Navigation and Communication on a Local Network

Start UWPlace on both nodes

Inject MessageTest to mnode8

Parent MessageTest agent spawns a child agent

Child agent hops to mnode9

Parent agent sends a message to child agent
Navigation and Communication over a Gateway

1: Inject to mnode8
2: Parent spawns child
3: Hop to gateway
4: Hop to uw1-320-20
5: Hop to uw1-320-21
6: Send message via gateway
7: Receive message
Using Java Sockets to Send and Receive a UWAgent

- Serialize UWAgent using ByteArrayOutputStream
- Create socket
- Using host name and port number, connect to recipient
- Write header to socket. Header contains method name (receiveAgent) to call at destination node, and amount of data remaining.
- Write serialized UWAgent to socket.
- Create ServerSocket
- Start thread
- Create Socket
- ServerSocket.accept waits for client connection
- Read / parse header
- Read serialized agent
- Call receiveAgent at local UWPlace to instantiate UWAgent and add it to list of local agents.
Why not RMI?

- `rmiregistry` process must be started and stopped manually
- The RMI communication layer must be configured properly
- Client on a gateway may send its public IP address to its server on a private network
- More control
Secure Communication

- Turned on or off from UWPlace command line
- Secure Socket classes are derived from Socket classes
- Use a certificate generated by keytool
// Create a ServerSocket or an SSLServerSocket
ServerSocket srvr = null;
if (uwplace.getIsSSL()) {
    SSLServerSocketFactory sslserversocketfactory = (SSLServerSocketFactory)
        SSLServerSocketFactory.getDefault();
    srvr = sslserversocketfactory.createServerSocket(portNum);
} else {
    srvr = new ServerSocket(portNum);
}
Secure Communication

// Create a Socket or an SSLSocket
InputStream in = null;
Socket skt = null;
if (uwP.getIsSSL()) {
    skt = (SSLSocket) srvr.accept();
} else {
    skt = srvr.accept();
}
in = skt.getInputStream();
Secure Communication

$ keytool -genkey -keystore UWAgentKeystore -keyalg RSA
Enter keystore password:
What is your first and last name?
  [Unknown]: Duncan Smith
What is the name of your organizational unit?
  [Unknown]: CSS
What is the name of your organization?
  [Unknown]: UW Bothell
What is the name of your City or Localility?
  [Unknown]: Bothell
What is the name of your State or Province?
  [Unknown]: WA
What is the two-letter country code for this unit?
  [Unknown]: US
Is CN=Duncan Smith, OU=CSS, O=UW Bothell, L=Bothell, ST=WA, C=US correct?
  [no]: y

Enter key password for <mykey>
  (RETURN if same as keystore password):
Monitor Commands

- as (Agent Status)
- kill
- suspend
- resume

--- Agent status ---
Number of agents: 3

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>MonitorTest</td>
<td>Ready</td>
</tr>
<tr>
<td>25</td>
<td>MonitorTest</td>
<td>Running</td>
</tr>
<tr>
<td>0</td>
<td>UWMonitorAgent</td>
<td>Ready</td>
</tr>
</tbody>
</table>
Class Name Collision

- Agents can carry additional classes
- Two agents may carry a class with the same name
- Testing UWAgent for this scenario
Questions?

http://depts.washington.edu/dslab/AgentTeamwork