The Creation of an Xgrid Controller in Java

Sean Colyer
2005-2006
TJHSST Computer Systems Laboratory
Period 1

Abstract
The creation of an Xgrid Controller in Java has a number of components ranging different branches of computer science. Xgrid is a distributed computer system created by Apple for use with it's Mac OS X. An agent for non Mac computers has already been created, but no one has created a Controller. The role of the Controller will be to be the central computer in a distributed network. Ideally once this project is complete it will be easy for any network to be quickly turned into a distributed network integrating Mac, Windows, and Linux computers.

Background
Apple has recently left the the PowerPC processor architecture it has been using for several years. It now uses Intel chips using the same architecture that Windows and Linux have been using for years (x86). With the announcement came the realization that it is not as difficult as it had been thought to convert PowerPC instructions into x86 instructions. With this new realization someone began to work on creating an Xgrid Agent for Java which has a number of advantages: Java runs on most operating systems without changes, Xgrid is an established network which can easily be adapted to, and it is relatively simple. With this advancement the ability for many computers to become clients or agents is possible, but the ability to create a Controller and utilize Mac, Windows, and Linux computers leaves a void. My project will fill this void, to do this I will experiment with distributed computing, processor management, network interfacing, and any other obstacles in my way.

Current Output:

```
Starting the server on a Linux computer
Available Processors: 1
Total Memory for VM: 2031616
Initializing StartChanneListener
Session established
Initializing StartChanneListener
StartChanneListener successfully intialized
Server is Listening
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist
version="1.0"><dict><key>identifier</key><string></string><key>name</key><string>agentRegistration</string><key
>payload</key><dict><key>addresses</key><array><string>fe80:0:0:0:211:85ff:fe0e:b84d%2</string></array><key>hos
tnames</key><array><string>tess.local.</string></array><key>agentCookie</key><string></string><key>agentName</k
ey><string>me</string><key>maximumCPUPower</key><string>2500</string><key>maximumTaskCount</key><string>1</stri
ng></dict><key>type</key><string>request</string></dict></plist>
type of message: request name: agentRegistration payloadDict: [dict: null]
Agent Registration: me  Running at: 2500Mhz
Message status: 2
Answer number: -1
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist
version="1.0"><dict><key>identifier</key><string>123</string><key>name</key><string>taskSubmission</string><key
>payload</key><dict><key>arguments</key><array><string>hello!!</string></array><key>command</key><string>/bin/e
cho</string></dict><key>type</key><string>request</string></dict></plist>
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist
version="1.0"><dict><key>identifier</key><string>123</string><key>name</key><string>taskSubmission</string><key
>payload</key><dict><key>taskRef</key><string>0</string></dict><key>type</key><string>reply</string></dict></pl
ist>
type of message: reply name: taskSubmission payloadDict: [dict: null]
Agent Replied.
```

Analysis and Conclusions
At this stage the Controller is listening for incoming connections, establishing the connections, and starting to communicate with them. The output is somewhat confusing to the untrained eye but in reality it is not that complex. The hardest part to understand is the XML messages which Xgrid uses for it's messaging. XML uses tags to encode messages in a way similar to HTML, these are interpreted using a SAX XML parser in Java. XML and BEEPcore were used in Java to create this implementation. BEEPcore is the method Xgrid uses for creating the network interface, so by using XML and BEEPcore the program can blend into networks because those are tools used by Apple in the creation of Xgrid. Just under the surface of the output is the management procedure of agents. The controller utilizes a HashMap of agents to contain Agents (which is based on the variables needed to keep track of an agent: channel of communication, address, name, processor power, busy status) and then referenced through the HashMap as messages are either received or sent. Using a synchronized receiveMSG class, the Xgrid controller will listen for incoming messages and not act until; this happens, this crucially helps the role of controller which depends on all the agents to help and works based on their feedback.

In the next stages of my project I will have to take this current implementation and extend it to encode processes into the Base64 strings used in Xgrid for sending processes to clients. Most of this should be done ahead of time by agents who are sending in requests to the controller which the controller will then redistribute. I will also need the controller to keep track of tasks that are having problems, such as disconnected agents. In the future the controller will redistribute more tasks.

Once the project is complete I will have to figure out the best way to create a simple installation that runs without user involvement and transparent to the user. The widespread availability and use of Java should make this relatively easy to implement in a wide range of environments. If this is successfully completed, the project could be used in widespread settings and the average times could be widely increased. Desktop PC's have enormous untapped potential. There are many situations where computers sit idle when they could be better suited to be contributing to a common good. In recent years clusters and distributed computing have made great strides, clusters now make a substantial portion of the top 500 supercomputers in the world.