Literature Review

More Efficient Serialization and RMI for Java - <http://haumacher.de/publ/parallel/p086.pdf>

* The paper goes on to explain that “polymorphism works on remote calls, i.e. every argument can be of its declared type or of any subtype thereof. Polymorphism is not possible in conventional RPM systems (including Corba, DCOM, and RMI-over-IIOP) because it requires dynamic loading and binding of stub code”.
* This highlights a reason for my chose of Java coding, as well as using Java RMI.

Java Remote Method Invocation – Distributed Computing for Java – Oracle - <http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138781.html>

* Using RMI allows for model behaviour to be passed as an object parameter to a client, which means that when a client receives an object, they are able to interact with that object as if it was the server interacting with the client (within reason, as a set-up can limit the methods published to the client through the use of an interface), as a pose to breaking an object down into a collection of the object state and passing the data to the client, meaning if the client does not have instruction on how to interpret the information to build initialise the object, the object behaviour is now lost.
* This highlights another reason for using Java RMI.

Push vs. Pull in Web-Based Network Management - <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=770671>