Assignment 3 – Short Answer Questions

Joel Schmuland & Jordan Mckenzie

# Question 1

## What are some of the advantages and drawbacks of using RMI versus network sockets for a client-server application?

As you would expect, there are many advantages of using RMI over network sockets which I will list below (RMI advantage list taken from Oracle [1]):

* **Object Oriented** – Network Sockets can be as well but you have to build it that way where as RMI is object oriented by default
* **Design Patterns** - beauty of using design patterns is that anyone that knows it can pick up what you are doing much easier then if you make it yourself.
* **Safe and secure** – don’t have to worry about making your code secure as RMI uses Java security mechanisms already in place
* **Easy to Write/Easy to Use** – because it is the same implantation used over and over it is easy to write and if anyone else looks at your code they can figure out what you are doing by knowing it uses RMI
* **Connects with Legacy System** – Don’t have to worry about what platform you are connecting to as long as it uses Java’s native method interface
* **Write once, Run Anywhere** – This motto from Java means that any RMI is compatible with any JVM
* **Garbage Collection** – No need to worry about object that are no longer referenced by any clients as RMI takes care of that for you
* **Native Parallel Computing** – RMI is Multithreaded. No need to write extra code for it.
* **Native Java Object Serialization** – no need to worry about breaking data down when passing it from one object to the next

Some of the disadvantages of using RMI over network sockets can be seen below:

* **Less Project specific –** network socket allows for a more granular approach when specifics are needed over a network
* **More Resource Intensive** – because network sockets are low level they are much lighter weight then using the full power of RMI

# Question 2

## What are some advantages and disadvantages of using Hibernate instead of JDBC for a data-connected application?

It is very similar to question one on a fundamental level. Asking this question is like asking why it is worth using any framework over writing the code out long hand. Here are some of the advantages of using Hibernate

* **Reduce Repeat Code -**  in hibernate you can save the object through persist(obj);
* **Minimize Code Changes –** There is a lot less changing that needs to be done if you want to add, for example, a new column to the DB
* **Database Independent –** Hibernate doesn’t really care what DB you use. It will allow you to use and switch between DB easily
* **Object Mapping –** instead of manually typing out how the DB works you can use annotations
* **Avoid Try Catch Blocks –** Hibernate changes all JDBC exceptions to unchecked exceptions

Some of the down sides of Using Hibernate over JDBC

* Much steeper learning curve – because you can access the DB quickly with JDBC its easy to get started but less efficient the deep you go

These ideas were pulled and re-evaluated from DZone’s page on Hibernate vs JDBC [2]

As you can see, there are many reasons one should choose both Hibernate and RMI when it comes to programming in Java