MPR/RBD sem3 project
(TODO title)

Intro

What we want to do will be basically similar to Sun's Project Wonderland. Very very much striped down of course. :)

http://en.wikipedia.org/wiki/Project_Wonderland

Scene Graph vs. OpenGL

In our first project JPong we used JOGL and Java3d's vecmath library to create our own, low level rendering system. This was good for practising pure OpenGL coding under Java. For our second project we decided to switch to a full scene graph based engine, mainly to practise working in a modern game development environments.

http://en.wikipedia.org/wiki/Scene_graph

Pure Java Scene Graph solutions

http://en.wikipedia.org/wiki/Java3d
http://en.wikipedia.org/wiki/JMonkey_Engine

This clip shows how Project Wonderland decided to switch from Java3D to JMonkey:

Part 8: Project Wonderland Technical Overview - New Graphics

“In the graphic system we are moving off Java3D, which has had a hard time keeping up with the latest developments of 3D rendering, to a scene graph technology called Java Monkey Engine or JME. JME is being used in a number of modern games, so we know it can do what we want graphically.”

And if Sun admits that JME is better then Java3D then who we are to argue? :)

JNI Scene Graph solutions

http://en.wikipedia.org/wiki/OGRE
http://en.wikipedia.org/wiki/Ogre4j

http://en.wikipedia.org/wiki/Irrlicht_Engine
http://sourceforge.net/projects/jirr/

This paper nicely summarises what is what in the world of Java JNI scene graph engines:
http://code.google.com/p/ogre4j/wiki/Whitepaper

OGRE is a very interesting engine and we were very tempted to use it. But, we already have some experience working with JNI from the JPong project, so this wasn't of so much interest to us. We decided not to use pure JNI engines for this project and stick with something more Java like. (JME of course also uses JNI to communicate with OpenGL native drivers, but it does this internally, behind the scenes.)

MMOG Midleware

http://en.wikipedia.org/wiki/Game_engine#Middleware

Project Wonderland used Project Darkstar as a MMOG middleware.
http://en.wikipedia.org/wiki/Project_Darkstar

Conclusion

Using JMonkey and Darkstar seems like a proper, modern way to implement MMO solution.

Java MMO game engine using Darkstar and JMonkey (youtube)

So what we want to do is write our own, very simple (M)MOG (2 users) server using Darkstar layout as a learning reference, and then use JMonkey to visualize it. Or simpler, every JME client will communicate with the external database directly throughout database wrapper library.