Implementing a GIS Using an OODB



Mary Garvey
Mike Jackson
Martin Roberts

Net Objects, October 2000

Structure of talk



- Overview of GIS
- GIS Problems
- Geographical Features
- Why Object-Orientation?
- GemStone Solution
- PJama Alternative
- Conclusions

What is a GIS?

- Wide number of viewpoints
- Relatively youthful and rapidly developing area
- *A GIS is an information system which stores spatially referenced data and makes provision for the spatial representation of data to be displayed graphically* (S.A. Roberts)
- First systems were mechanical using punched cards

GIS Problems

- Earlier systems failed because slow and inefficient
- Others were badly designed - designers failed to appreciate technical problems involved in manipulating spatial data, e.g., Land Use and Natural Resources (LUNR) system
- Satellite data, e.g., Landsat, increased the need for more efficient systems

Geographical Features

- A good design integrates spatial and aspatial data
- Spatial:
  - Points: crossroad, traffic signals, bridge
  - Lines: road, river, railway
  - Area: town, lake, car parking
- Scaling causes problems
- Aspatial:
  - attribute, statistical or non-locational data
  - spatial entities can have associated aspatial data
- Some systems consider spatial to be more important than aspatial

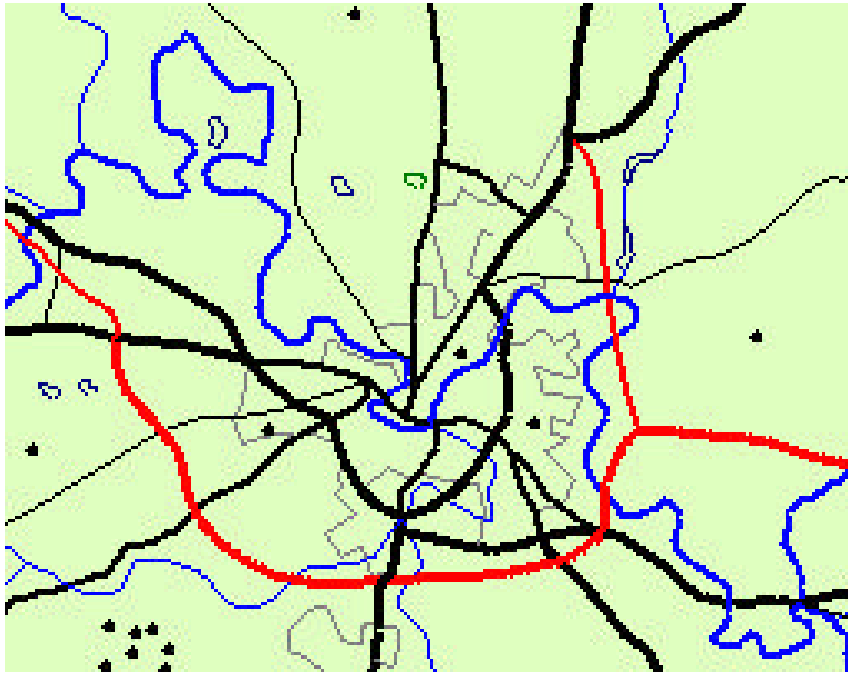
Geographical Functionality

- GIS not just about data structures
- Must provide functionality required by the user, e.g.,
  - What is adjacent to ...?
  - Show all houses 100 yds from the A6
  - Shortest route
  - Spatial patterns
- Can include aspatial data:
  - What is the total number of planning applications in a certain area?
  - What is the total rateable value along a proposed new road?

Why Object-Orientation?

- Ability to represent complex data
  - cartographic data does not fit easily into traditional record structures
- Encapsulation
  - ability to capture the behaviour with the data structures
- Capturing semantics
- OODBs behave like object-oriented languages however the objects are persistent

Map with some data



Sample Data

```
1LINE      230                          2
   342380.000000000   321300.000000000       .000000000
   342320.00000000   321620.000000000        .000000000
2LINE      230                          2
   342440.000000000   321060.000000000        .0000000000
   342380.000000000   321300.000000000        .0000000000

10AREA     210                          19
   350820.000000000   307580.000000000        .0000000000
   350847.343800000   307603.218700000        .0000000000
   350865.156300000   307632.093700000        .0000000000
....
   350749.812500000   307577.031200000        .0000000000
   350820.000000000   307580.000000000        .0000000000

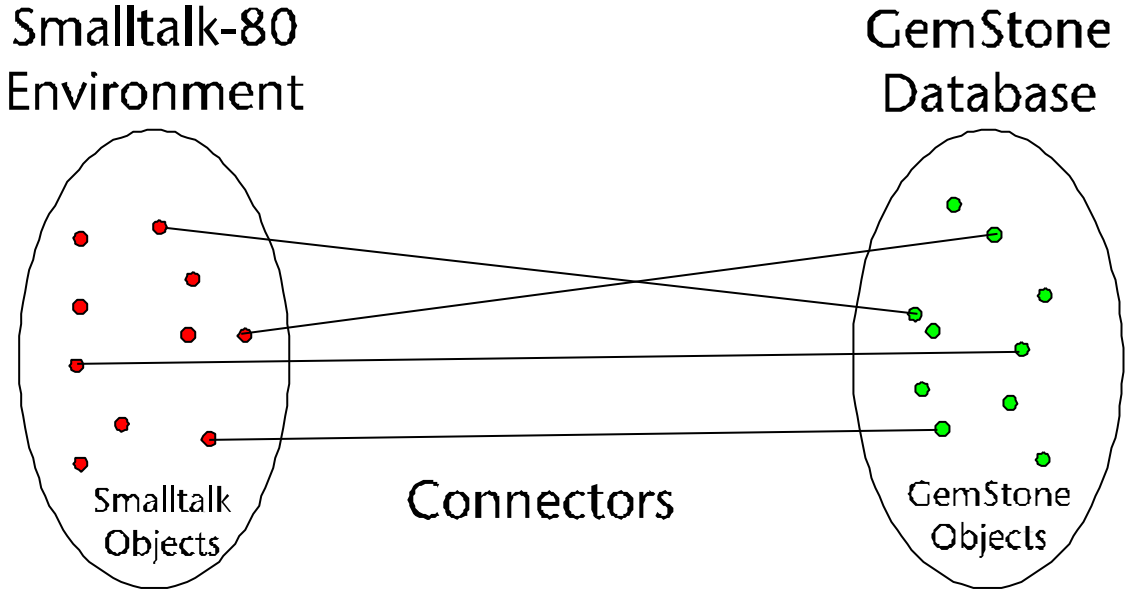
1POINT     116                          1
   348033.754070001   318661.908200000        .0000000000
2POINT     31                           1
   342271.231610000   317325.488280000        .0000000000
```

Why GemStone

- Systems initially evaluated:
  - Postgres
  - Geo++
  - Ode
- GemStone offered a choice of interfaces:
  - GeODE
  - C++
  - Smalltalk
  - C
- GemStone and Smalltalk interface (GSI) offered most flexibility and more extensive set of built-in classes

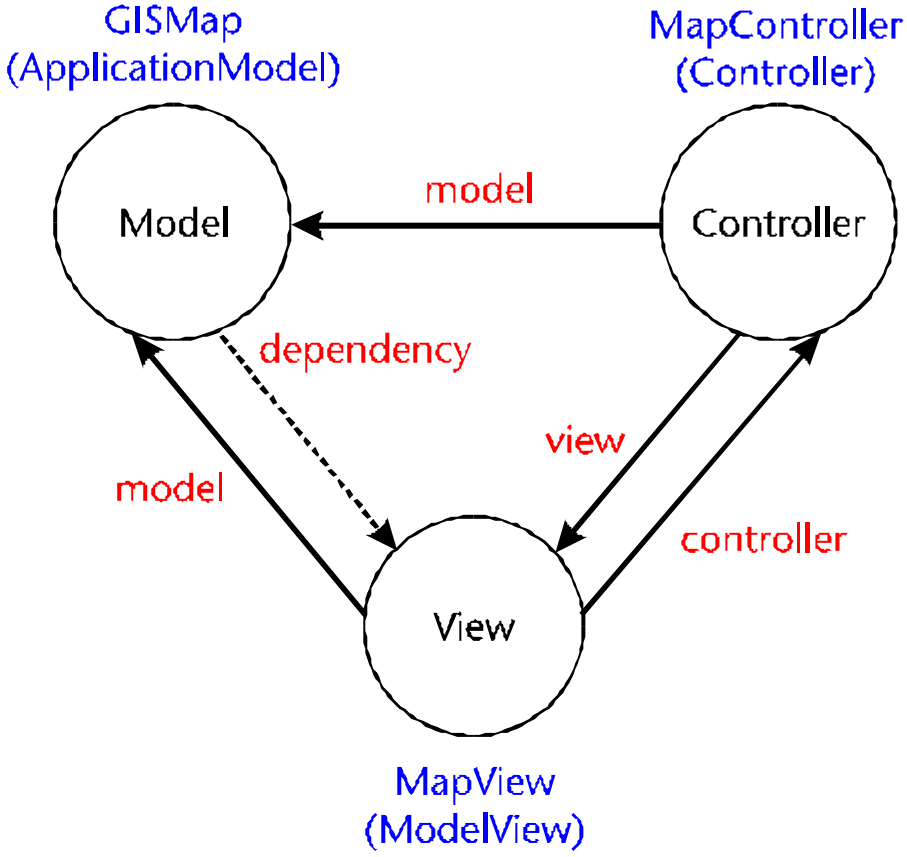
Smalltalk Environment

- Two environments within GSI:
  - Smalltalk-80
  - Smalltalk-DB (formerly known as Opal)
- Smalltalk-80 richer class library
- Use Connectors to communicate with each other



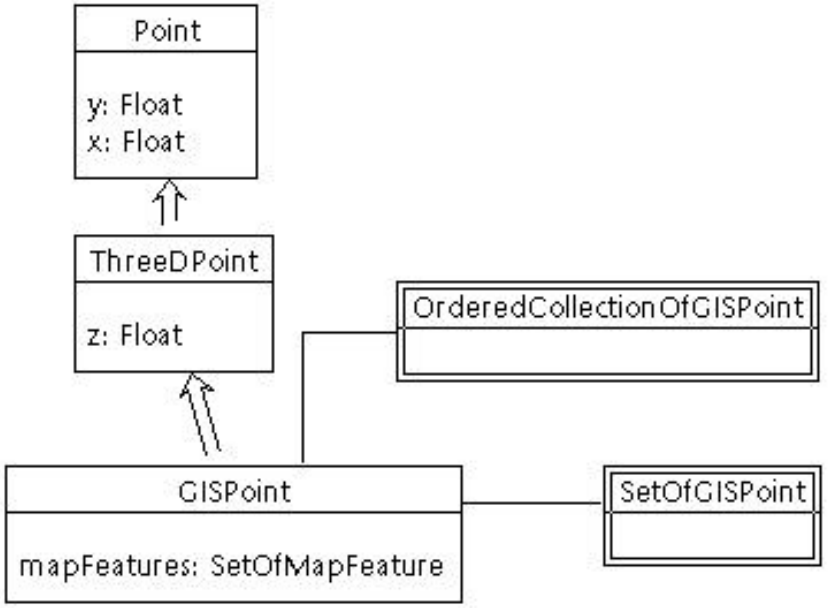
MVC Architecture

- Model-View-Controller (MVC) provides powerful architecture for building user interfaces



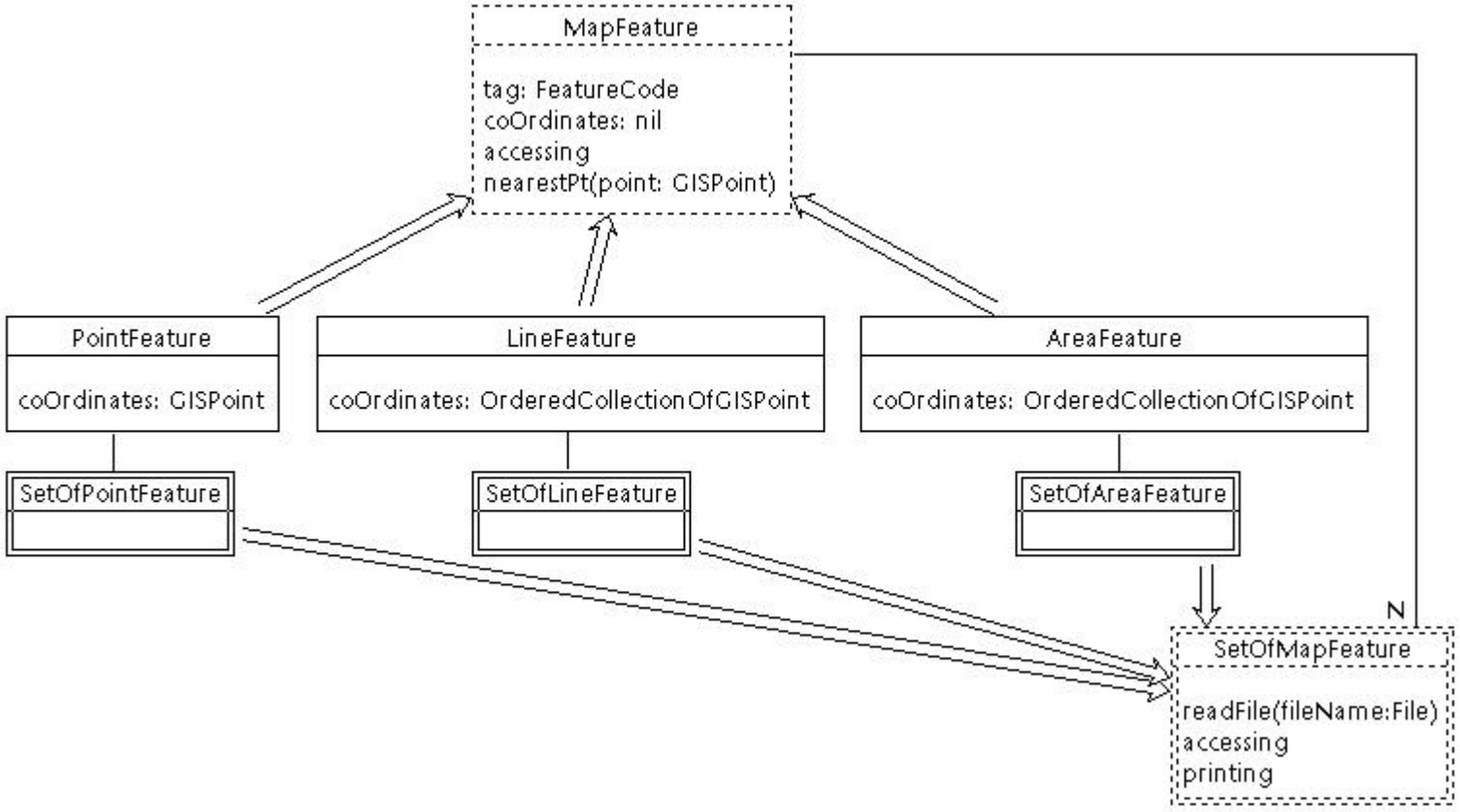
GIS Schema

- GISPoint crucial to overall schema
- 3D point plus reference to all features at that point



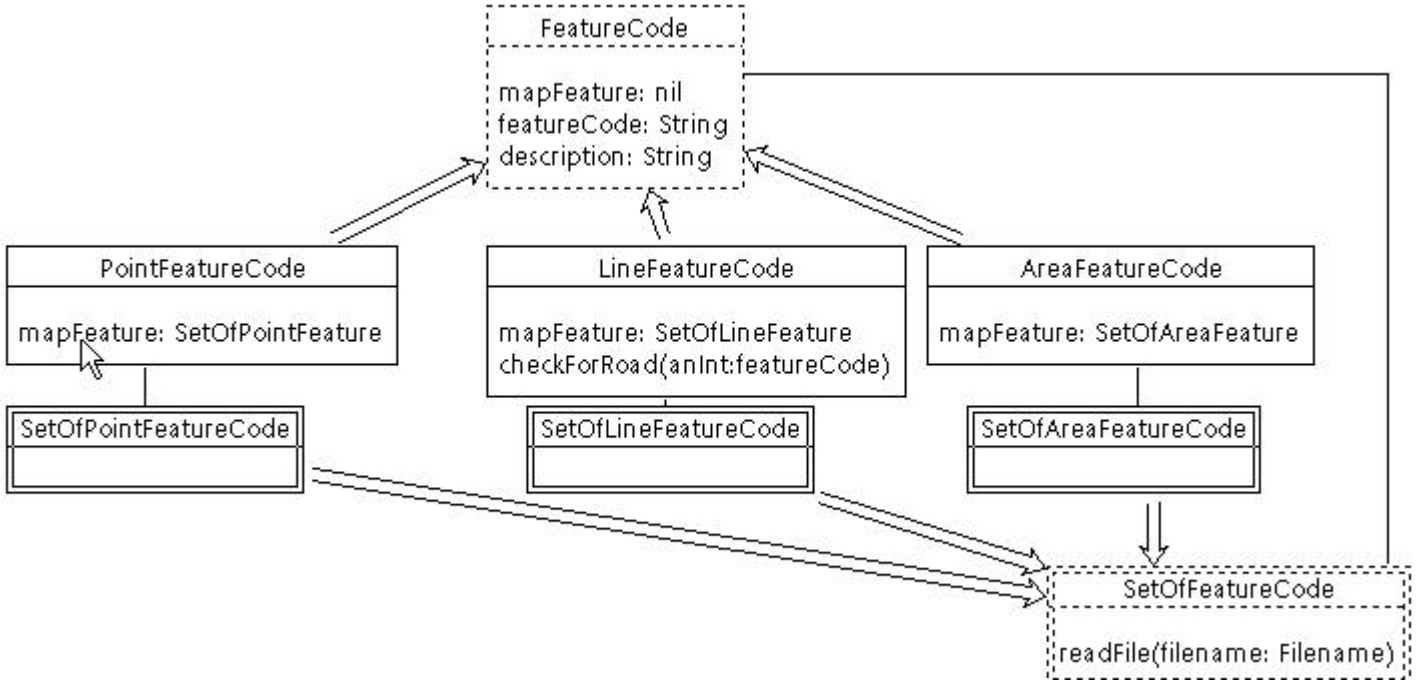
Map Features

- Holds the map data



Feature Codes

- Contain feature types, e.g., A-road, Motorway and references to all points of that type



Map

- The map object links everything together



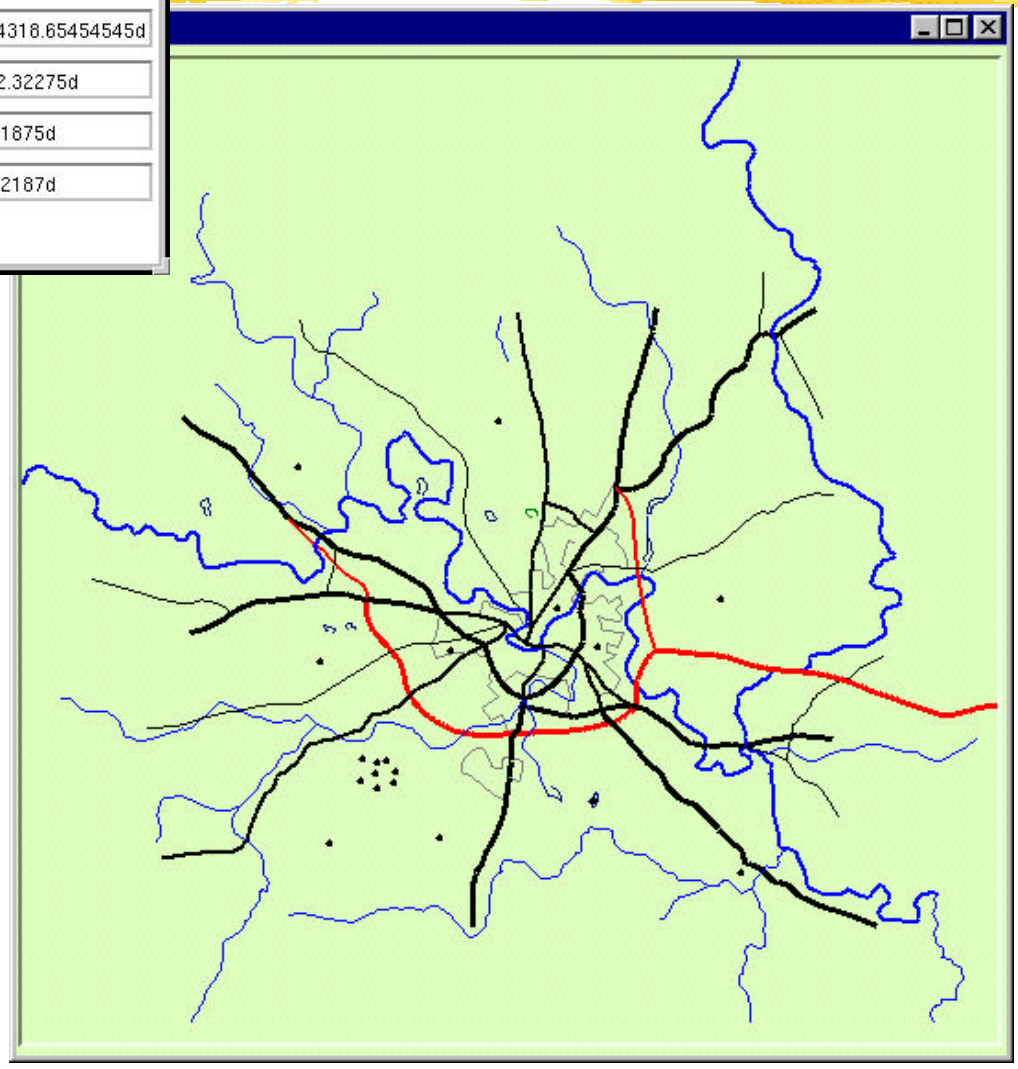
GemStone Map

Map Details

System Load Data Functions Sizing

Current Point	174@271	GIS Point	343553.91465677d@314318.65454545d
Pt. Description	GARDEN OR GROUNDS	Nearest Point	342914.25944d@311732.32275d
Line Description	A-ROAD SINGLE C/W UNDE	Nearest Line	343561.0312d@314326.1875d
Area Description	LAKE OR RESERVOIR	Nearest Area	343725.6875d@312843.2187d

Line nearest



Advantages - GemStone

- Smalltalk has a rich set of relevant classes
- MVC - key to fast application development
  - Easy to colour code the roads etc.
- Ability to create dynamic classes for aspatial data
  - Use of the meta-classes (Class, Behavior and ClassDescription) allows the user to build their own classes
  - The GIS system generates standard *setters* and *getters* methods
- Effort required to implement system less than if used a relational system

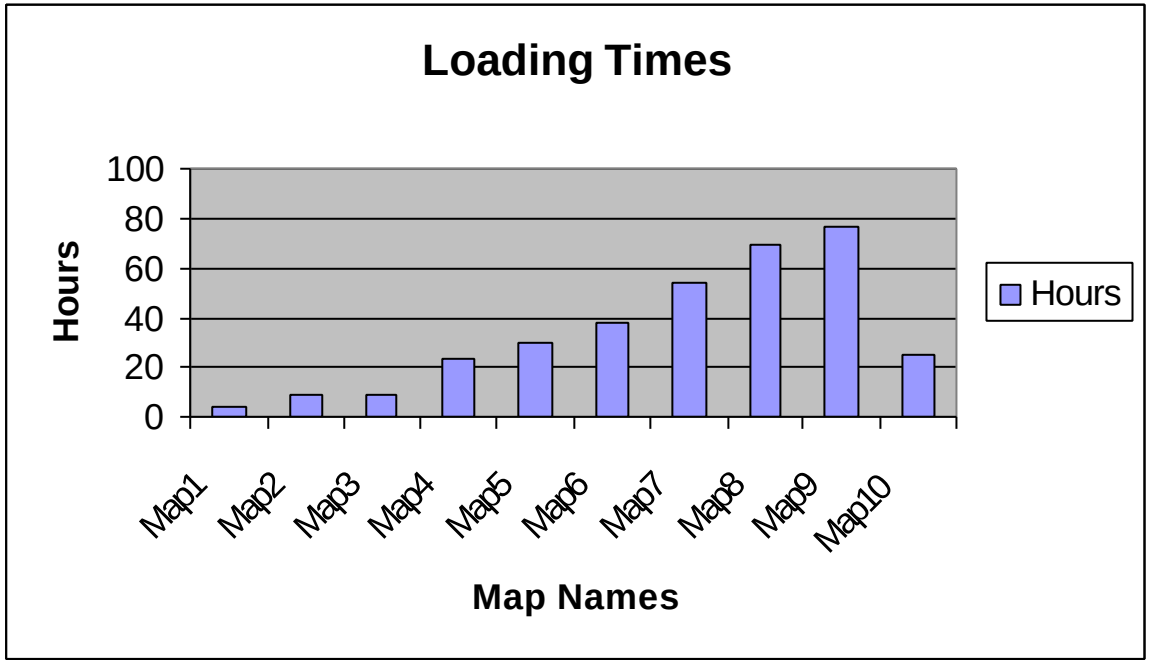
GemStone - Loading Data



- Initially loaded data into Smalltalk environment, then committed to GemStone
- Loading a small dataset caused no problems, e.g., a dataset of 450 kb took 30 seconds to load and commit.
- Problems started occurring when extra classes were added for network analysis and larger datasets were used
- Forwarders had to be used to cope with the larger datasets
- Even loading data directly into GemStone via Topaz is slow

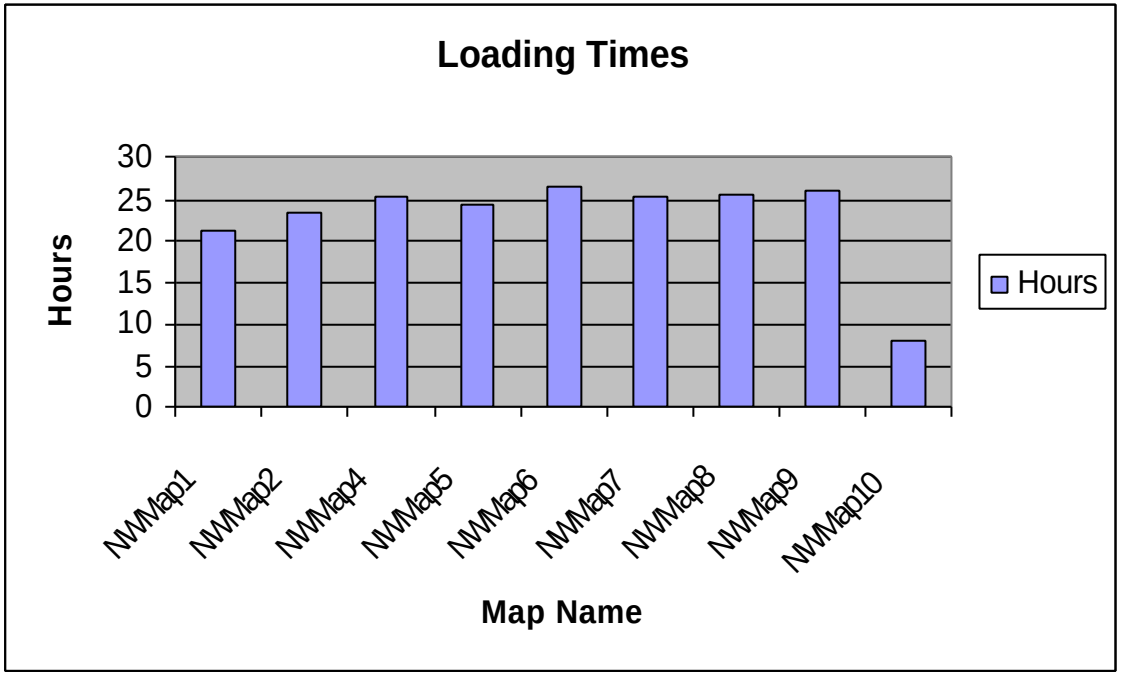
GemStone Load Figures

- Split a 5.6 Meg dataset containing roads in the North West into 10 maps, approximately 10,000 lines each (average 1240 segments)
- At one point took 340 hours in total to load



Load Figures Continues

- Found if committed more often, load time decreases but still unacceptable overall



Cold and Warm Starts

- Once loaded the problems are not over
- Cold start can take up to 30 minutes on the North West data
- Once loaded warm starts are fairly quick
  - Can get a lot of garbage collecting which slows down performance
- Speed of functions such as nearest point and shortest path however execute quickly

Java & PJama Version

- Different solutions looked at to speed up the GemStone version
- Alternative was to rewrite the GIS in Java
- PJama provides the persistence for the data
- Programs written without any regard for how the data is actually permanently stored
- Same schema has been used
- Java also supports MVC

PJama Advantages

- Porting the schema proved straightforward
- Little amendments needed to the Java code, e.g.,

```
import java.util.*;
import org.opj.*;

public class MapDictionary
{
  static { OPRuntime.roots.add(MapDictionary.class); }
  private static Hashtable mapList = new Hashtable();

public Object getPRoot(String name) {
     return mapList.get(name);
  }
public void newPRoot(String name, Object root) {
     mapList.put(name, root);
  }
/* other methods ... */
}
```

PJama Continued

- To commit the database straightforward:

  OPRuntime.checkpoint();
- Performance improved - loading data:
  - 3 secs for Shrewsbury data
  - 30 sec for North West data
  - cold starts also less than a minute
- Proved to be too persistent! From the MVC architecture only the model should be persistent.
- To prevent view being saved must also use - *nowindowresume* parameter and windows must be disposed of when no longer required

Sun Hardware

- Both systems not run on similar platforms
- PJama Version 1.6.5
  - Ultra-10
  - 2 X 300 MHz CPU
  - 512 MB RAM
- GemStone Version 5.1.2
  - Sparc Station 20
  - 2 X 75 MHz CPU
  - 448 MB RAM
- In process of moving both onto the same server to give a fairer comparison and move to GemStone 5.1.5

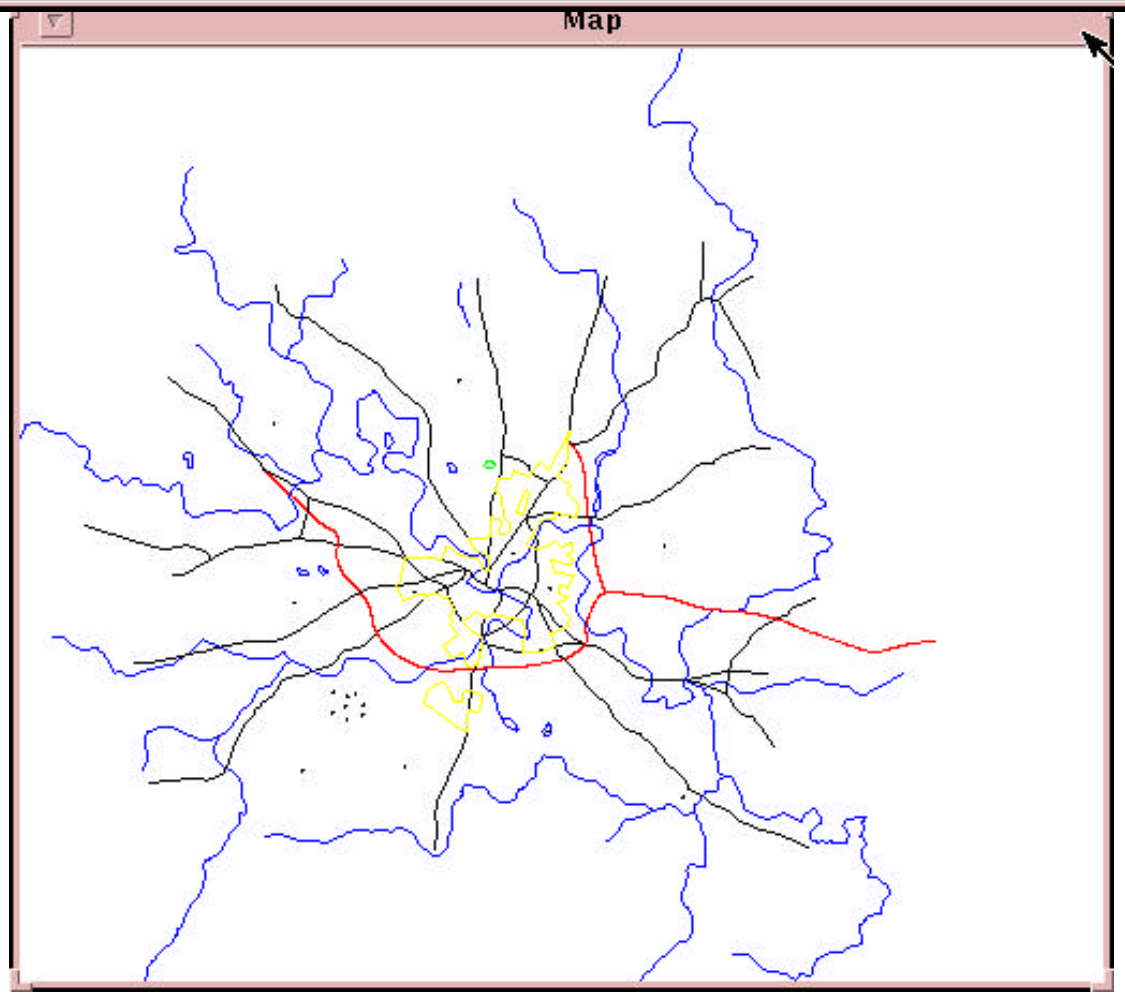
PJama Map

Map Details

File GIS Functions Zoom

Current Point	.Point[x=200,y=10	GIS Point	6666.02024406375	Nearest Point	).3, 313514.22, 0.0	Nearest Line	10.0, 316540.0, 0.0
Nearest Area	47, 316539.62, 0.0	Point Description	POINT	Line Description	RIVER 6TH ORDER	Area Description	LAKE OR RESERVO

Map



Future Work



- Add further GIS functionality to PJama version, e.g., point-in-polygon
- Link spatial data to aspatial data
- Add ability to allow users to dynamically create their own classes
  - This may be more difficult in Java than Smalltalk

Conclusions



- Applications are quicker to develop using Object-Oriented technology compared to achieving similar results in a relational or even object-relational system
- MVC architecture key to fast development times
- Performance in GemStone not lived up to expectations
- PJama proving to be a more viable alternative