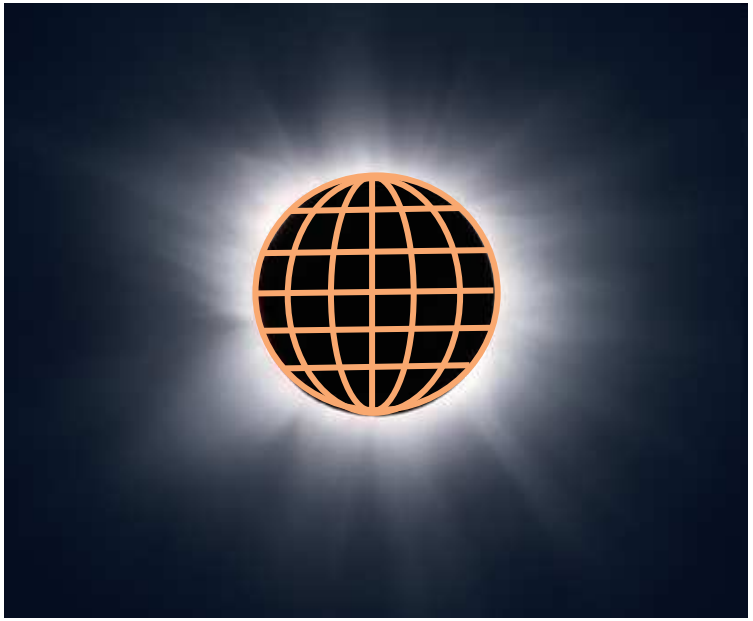
Harman International

Europe on a Disk

Geodata Processing with Eclipse and OSGi

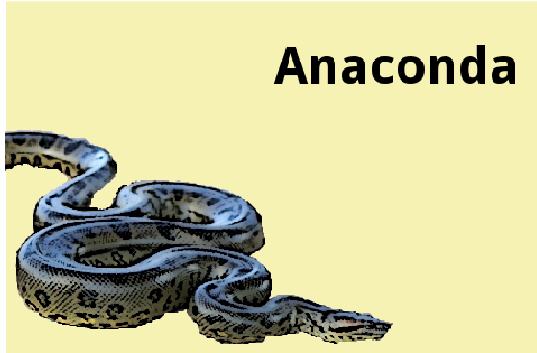


Harald Wellmann

10 Nov 2008

Overview

- Past and Present of Navigation Data Processing
- Anaconda: The Future



- Our usage of OSGi and Eclipse

BECKER


In the Middle Ages,
Earth was a disk.

Today, we know better...

…and it‘s our job to put it on a disk again

What is a Map Compiler?

- A Map Compiler compiles data, not programs.
- Raw geo-spatial data is compiled into a compact binary format.
- Data gets enriched, not just transformed
- ...to make the result usable in a car navigation system.

Our Map Compiler Today

- A large distributed application
  - written in C++ and C
  - uses proprietary file managers and database engines
- Output: proprietary binary formats
- Input: GDF

GDF

Geographic Data Files

70's Style Geodata

A Look at GDF

```
01NAVTEQ                GDF       3.0 070830                    1       1         1
00ISO-8859-1        10000000207   1(C) NAVTEQ NORTH AMERICA, LLC] SV12.70,FV05.71
000,DV2.4.3,CT20:53,DVNGERMANY G1 07303,LNG3G1GFP ] SV12.70,FV05.70,DV2.4.3,CT21
000:21,DVNGERMANY G2 07303,LNG3G2GFP ] SV12.70,FV05.70,DV2.4.3,CT19:16,DVNGERMA1
00NY G3 07303,LNG3G3GFP ] SV12.70,FV05.70,DV2.4.3,CT19:53,DVNGERMANY G4 07303,L1
00NG3G4GFP ] SV12.70,FV05.70,DV2.4.3,CT20:02,DVNGERMANY G5 07303,LNG3G5GFP ] SV1
0012.70,FV05.70,DV2.4.3,CT20:00,DVNGERMANY G6 07303,LNG3G6GFP ] SV12.70,FV05.701
00,DV2.4.3,CT19:11,DVNGERMANY G7 07303,LNG3G7GFP ] SV12.70,FV05.70,DV2.4.3,CT191
00:42,DVNGERMANY G8 07303,LNG3G8GFP ]  LNEG0GFP                                   0
0201              0000000207                                                     0
03ALBUM_ID  10N      OBL                            UNIQUE ALBUM IDENTIFIER      0
03AREA_ID   10N      <S>   1          2E32         AREA FEATURE IDENTIFIER      0
03ATT_DESC  * G      <S>                            ATTRIBUTE DESCRIPTION        0
03ATT_DIR   1 G      <S>                            VALIDITY DIRECTION           0
03ATT_TYPE  2 G      OBL                            ATTRIBUTE TYPE CODE          0
03ATT_VAL   10G      <S>                            ATTRIBUTE VALUE              0
03CHAR_SET  10G      OBL                            CHARACTER SET USED           0
03COMF_ID   10N      <S>   1          2E32         COMPLEX FEATURE IDENTIFIER0
03COMMENT   * G      <S>                            COMMENT                      0
03CONT_VOL  1 N      OBL   0          1            VOLUME CONTINUATION MARK     0
03CORI_DATE 6 N      <S>                            COPYRIGHT DATE               0
03CORI_OWNER* G      <S>                            COPYRIGHT OWNER              0
03CREA_DATE 6 N      <S>                            CREATION DATE                0
03DASET_ID  10N      OBL                            DATASET IDENTIFIER           0
03DATA_TYPE 2 A      <S>                            DATA TYPE                    0
03DATA_UNIT 3 A      <S>                            DATA UNIT                    0
03DATA_USE  2 G      <S>                            FIELD CLASS                  0
03DEC_ANNUAL5 I      <S>   0          +4000        ANNUAL CHANGE                0
03DEC_DATE  6 N      <S>                            VALIDITY DAT                 0
03DEC_VALUE 5 I      <S>   0          +4000        MAGNETIC VARIATION           0
:_
```


New Data Formats

- Input: RDF
- Relational Data Format from NAVTEQ

- Output: NDS
- Navigation Data Standard
- A vendor independent binary format for the next generation of Car Navigation Systems

Navigation Data Standard
PSF Physical Storage Format

A Look at NDS

- Industry standard
- Formally specified
  - using DataScript
- Supports incremental updates

Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe

http://k332.hbi.ad.harman.com:8080/job/anaco

DataScript Package: psi.all

- **all packages**
- psi.all
- psi.amd.amd
- psi.amd.img

- CompleteImplic
- CompleteNam
- ConnectLinkTy
- ContainedNam
- ContentLayerT
- ContentLayerT
- ContinuedTurn
- *CoordWidth*
- CoordXY
- CoordXYDiff
- CoordXYDiffWi
- CoordXYOffset
- *DataModelVer*
- DataModelVers
- DataQuality
- DateRangeOfY
- DateTime
- DayOfYear
- DaysOfMonth
- DaysOfWeek
- DecryptionInfo
- DecryptionInfo
- DescriptiveNar
- DestLaneSpec
- DestLanes
- *DetailedCityMa*

psi.common.geometry

Sequence CoordXYDiff

Remaining vertices of a line feature in difference encoding. Each vertex is specified by an offset from the preceding vertex. The first vertex of the line is defined by the context and is not part of this type.

The vertex coordinates are defined recursively by

```
v[i].x = v[i-1].x + (1-2*signX) * dx
v[i].y = v[i-1].y + (1-2*signY) * dy
```

Param: `numVertices` -> number of vertices contained in this type

```
CoordXYDiff(uint16 numVertices)
{
    bit<4>                          numBits;
    CoordXYOffset (numBits + 1) offsets[numVertices];
};
```

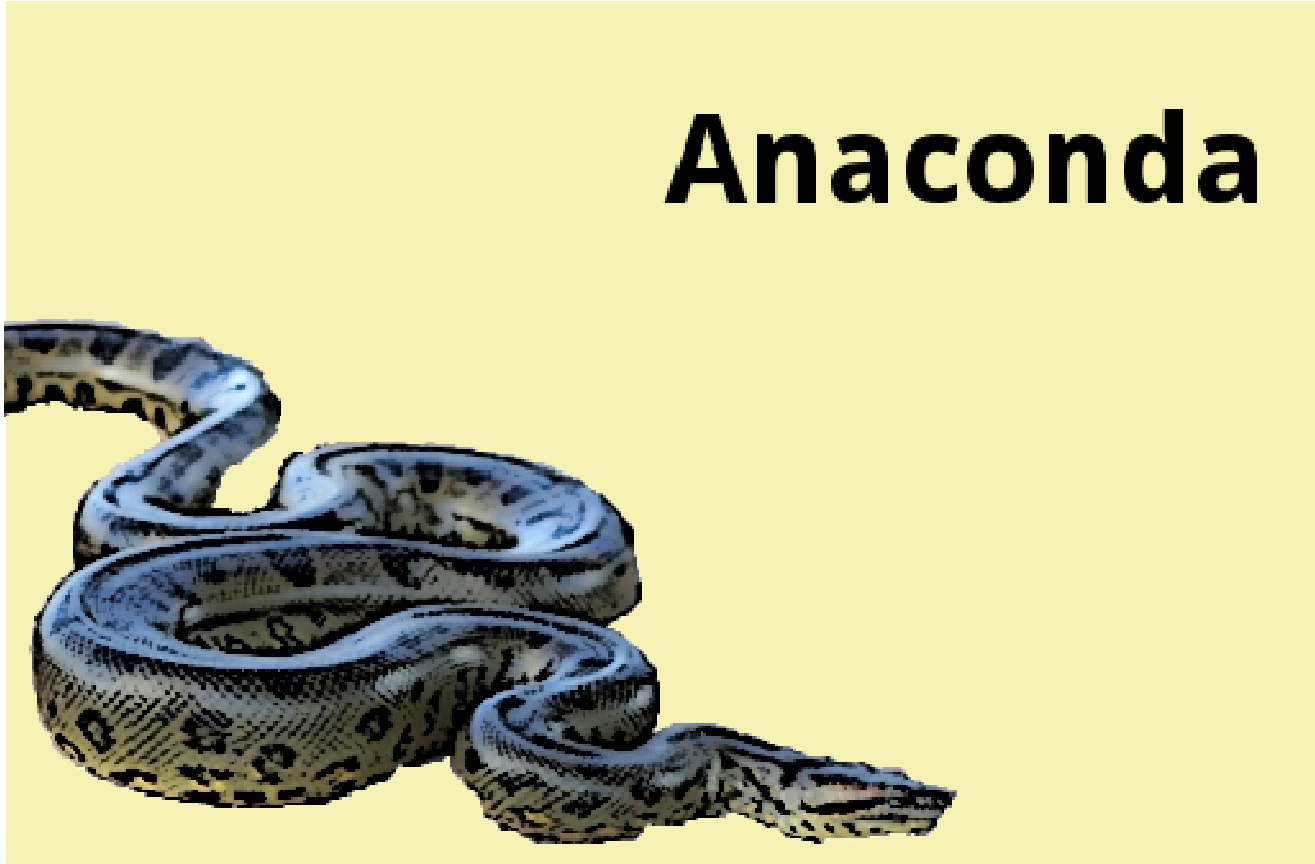
Member Details

numBits:
Adding 1 to this value yields the number of bits for the absolute value of x and y coordinate offsets. This value is in the range 1..16.

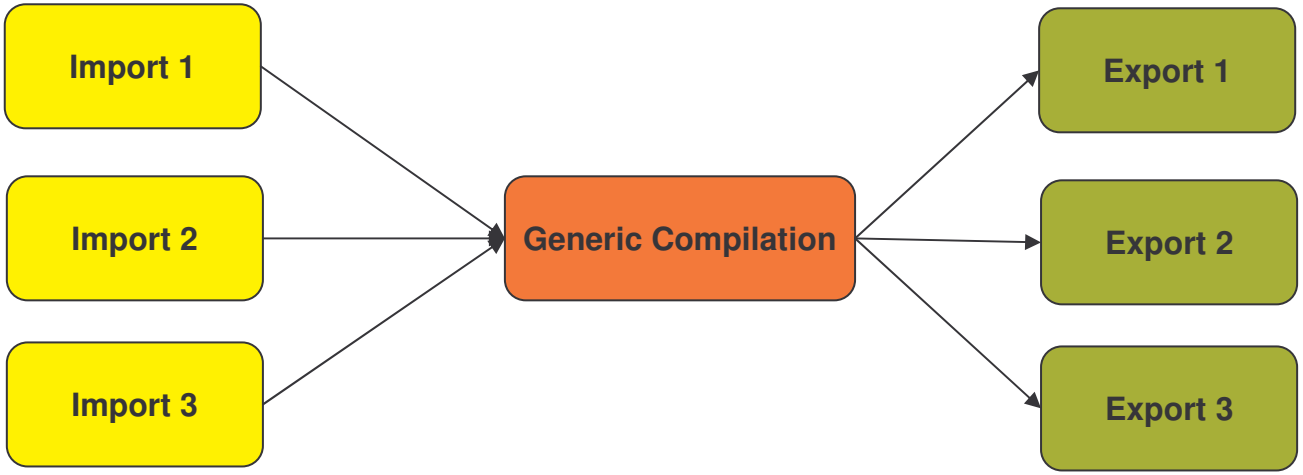
offsets:
List of coordinates for remaining vertices.

Fertig

A New Architecture for Compiling Navigation Databases

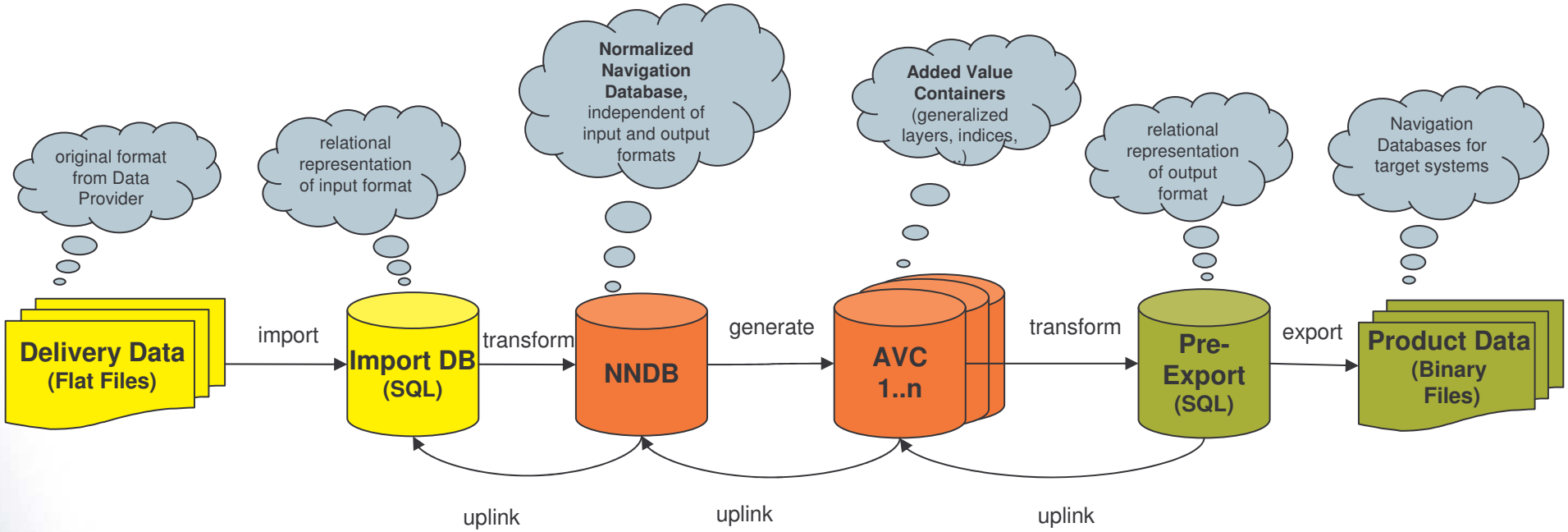


Data Flow



- Anaconda supports multiple input and output formats.
- Compilation is largely decoupled from these formats.
- Data Supplier format changes only affect the import part.

Generic Compilation Process



Supplier-dependent formats

Generic Anaconda formats

Product-dependent formats

Architecture Requirements

- Use standard database technology
- Use standard libraries and APIs wherever possible
- Enable distributed compilation
- Enforce component architecture with controlled dependencies

Implementation Language

- The implementation language is Java
  - Java 6.0 is required

- Motivation:
  - Developer productivity
  - A wealth of libraries for all generic and many domain-specific tasks
  - No need to used embedded approaches in a server environment
  - Easy building
  - Platform independence

Development Environment

- Anaconda uses the Eclipse Integrated Development Environment.
  - Eclipse 3.4 (Ganymede)

- Motivation:
  - Wide-spread, open source
  - easy integration of other tools
  - Support for OSGi-based development

OSGi: Component Model

- Anaconda uses the OSGi component model.
- Each Anaconda component is an OSGi bundle.
- A bundle declares its dependencies on other bundles in its manifest.
- Classes from other bundles are not visible at run-time, unless the dependency is declared explicitly.

OSGi Implementation

- Anaconda uses Eclipse Equinox
- Equinox has an extension called buddy class-loading enabling class loader callbacks:
  - `Class.forName(userProvidedLibraryExtension);`
- This is required by certain third-party libraries used by Anaconda.

- Thus, Anaconda will currently not work with other OSGi implementations.
- This would not be required if all third-party libraries were OSGi compliant.

OSGi vs. Eclipse

- Anaconda is a pure OSGi console application.
- Eclipse is just the IDE.
- Anaconda does not depend on any Eclipse components
- ...except `org.eclipse.osgi`

Getting Started

- Start a worker thread in the bundle activator.
- Make sure the thread survives the startup phase.
- Do not call long running or blocking methods in the bundle activators.

```
public void start(BundleContext ctx) throws Exception
{
    log.info("starting bundle com.harmanbecker.anaconda.main");
    Thread worker = new Thread(this, "Anaconda Main Thread");

    worker.setDaemon(false);
    worker.start();
}
```


Running

- The job to be executed is a `Runnable`, registered as an OSGi service.

- The worker reads a job name from the config file,
- gets a matching service from the service registry
- and runs it.

```
public void run()
{
    String configFile = System.getProperty("anaconda.config");
    log.info("reading Anaconda configuration from " + configFile);

    Configuration config = Configuration.getInstance();
    String className = config.getProperty(JOB_START_KEY);
    JobRegistry.getInstance().runJob(className);
}
```

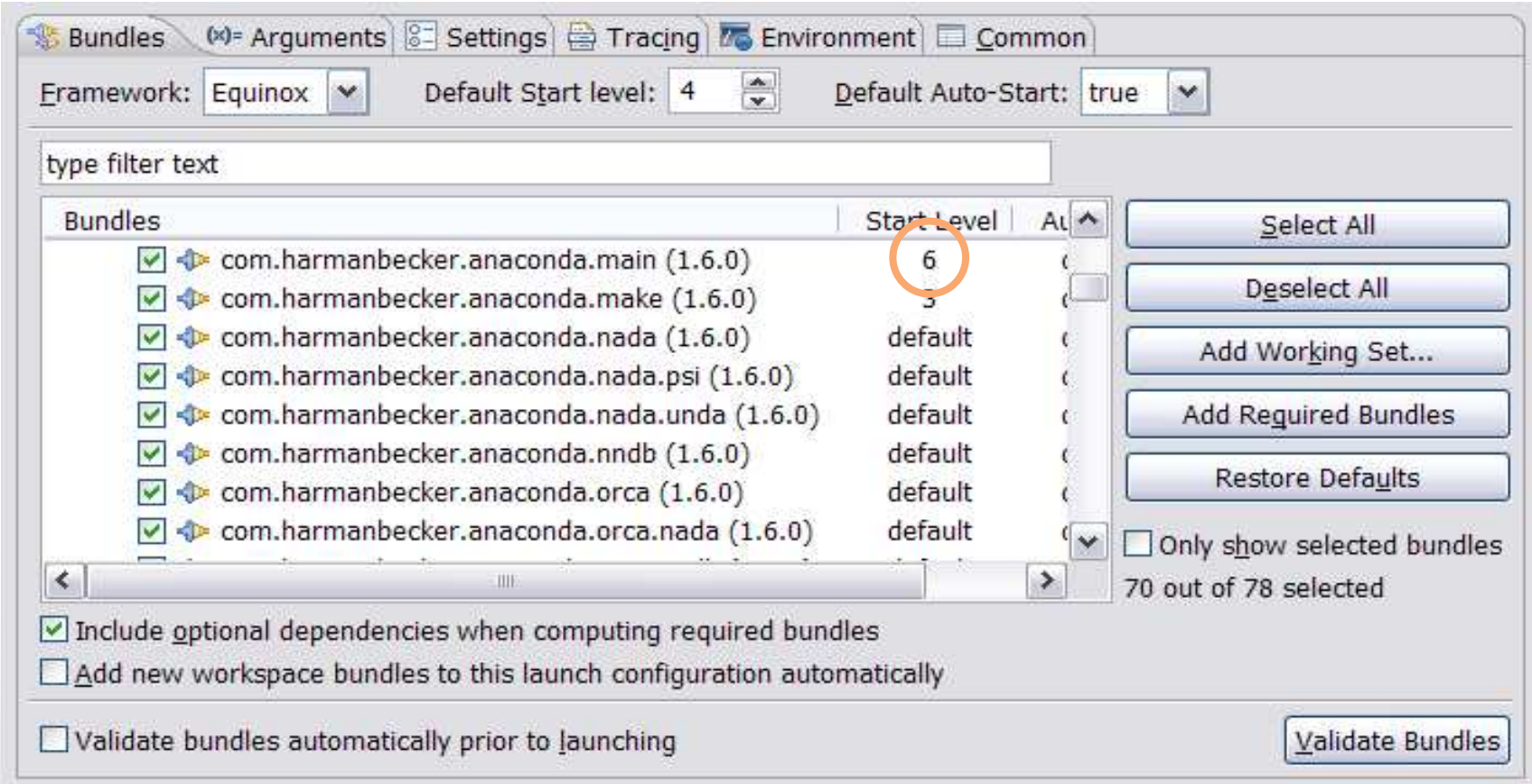

Static Services

- We do use OSGi services...
- ...but none of the dynamics.

- All services are registered during Framework startup.

- Bundle start levels ensure that a service is registered before it gets used.

Start Levels



- The main bundle is started last.

Stand-alone Deployment

- Anaconda is a console application running in an OSGi environment.

- Anaconda is installed in two folders:
  - `plugins:` containing the OSGi framework and all bundles
  - `configuration: config.ini` listing bundles to be activated

- Command line for launching Anaconda

```
java <VM options>
    -jar plugins/org.equinox.osgi.jar
    -configuration <config folder>
```


Equinox Configuration File

- Mainly a list of bundles with start levels

```
# Enabling bootdelegation ensures that JRE classes
# that do not belong to the standard OSGi runtime environment
# will be loaded from the JRE.
org.osgi.framework.bootdelegation=*

# Do not run any Eclipse application, we are in pure OSGi mode.
eclipse.ignoreApp=true

# List of bundles to be installed and started.
osgi.bundles=\
com.harmanbecker.anaconda.antlr.fragment_1.6.0.jar, \
com.harmanbecker.anaconda.conductivity_1.6.0.jar@start, \
com.harmanbecker.anaconda.core_1.6.0.jar@start, \
com.harmanbecker.anaconda.freetext_1.6.0.jar@start, \
com.harmanbecker.anaconda.generalize_1.6.0.jar@start, \
com.harmanbecker.anaconda.hibernate.fragment_1.6.0.jar, \
com.harmanbecker.anaconda.jump.nada_1.6.0.jar@start, \
com.harmanbecker.anaconda.jump.psi_1.6.0.jar@start, \
com.harmanbecker.anaconda.jump.unda_1.6.0.jar@start, \
com.harmanbecker.anaconda.main_1.6.0.jar@6:start, \
com.harmanbecker.anaconda.make_1.6.0.jar@3:start, \
com.harmanbecker.anaconda.nada.psi_1.6.0.jar@start, \
```


Benefits of OSGi

- Bundles give structure to your application.
- Easy to use, if you play by the rules.

But...

Harman International

L'enfer,

c'est les autres.

Jean-Paul Sartre

Third-Party Dependency Hell

- Third-party libraries come as plain old JARs, not as OSGi bundles.
- Libraries depend on other libraries.
- Libraries have dependency cycles.
- Version conflicts in transitive dependencies.
- Libraries use `Class.forName()`

Ways out of Hell

- Get ready-to-use OSGified libraries from SpringSource Enterprise Bundle Repository.
  - Caveat: Some manifests are broken or do not work in every environment.
- Use Maven to manage your dependencies.
- Use buddy policies or fragment bundles to work around classpath issues.
- Rebuild libraries from source.
- Repackage a JAR with a new manifest.
  - `maven-bundle-plugin` helps

NO CYCLES

True or False:

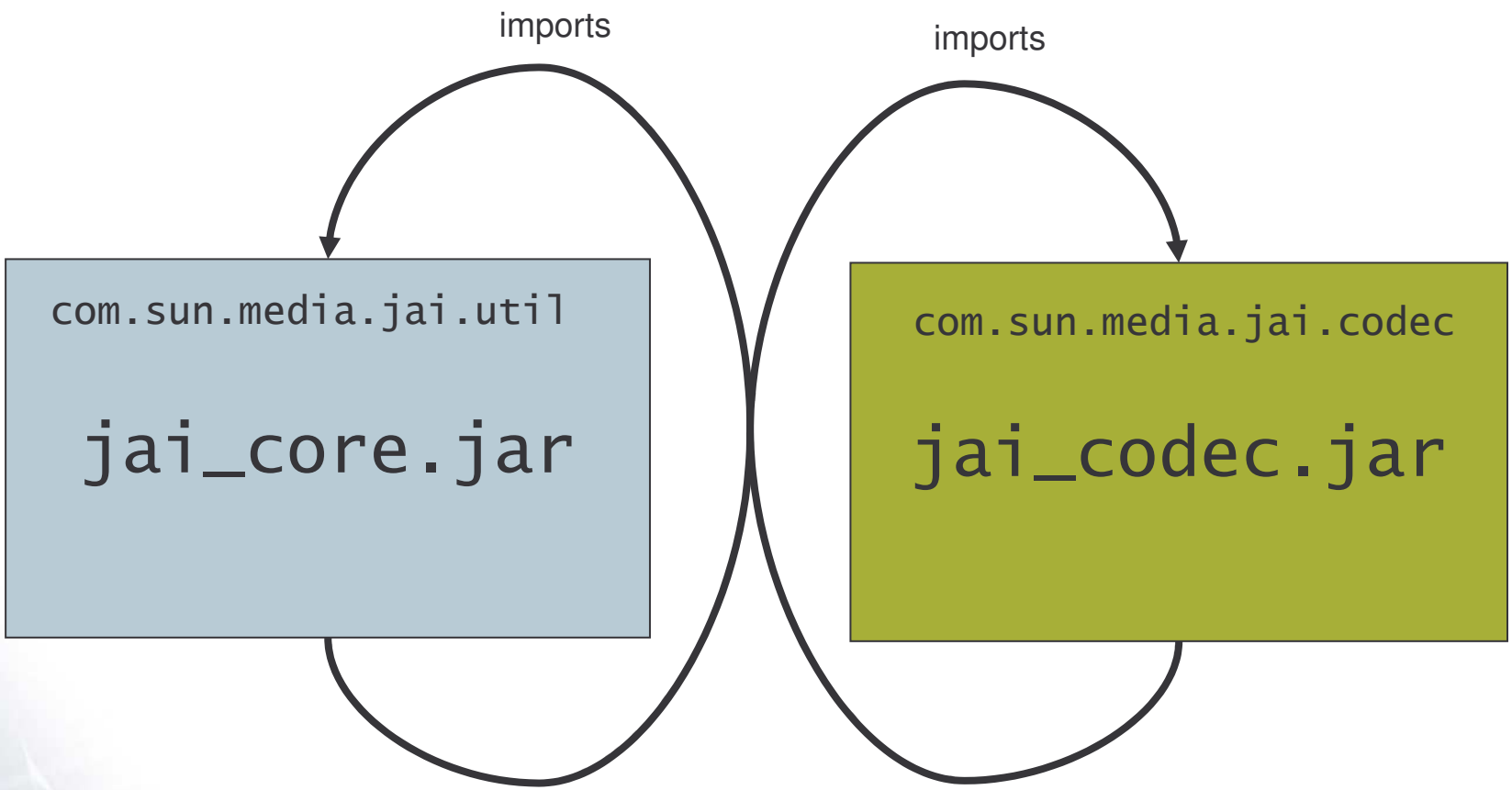
Dependency cycles are a Bad Thing.

OSGi does not allow dependency cycles.

- n-1 Bundles from an n-cycle remain unresolved
- All bundles get resolved on installing the last one

Eclipse has restrictions on plugin cycles.

Sun's Sins



Eclipse Cycle Woes

- Eclipse PDE Batch build terminates when it finds a dependency cycle in your target platform.
- ...even though it does not have to build these bundles.

- There is partial relief:

```
allowBinaryCycles = true
```

- undocumented `build.properties` option in Eclipse 3.4
- to become visible on the UI in Eclipse 3.5

A View on the Map Database

- Compiling is not enough.
- You want to visualize the results.

Viewers

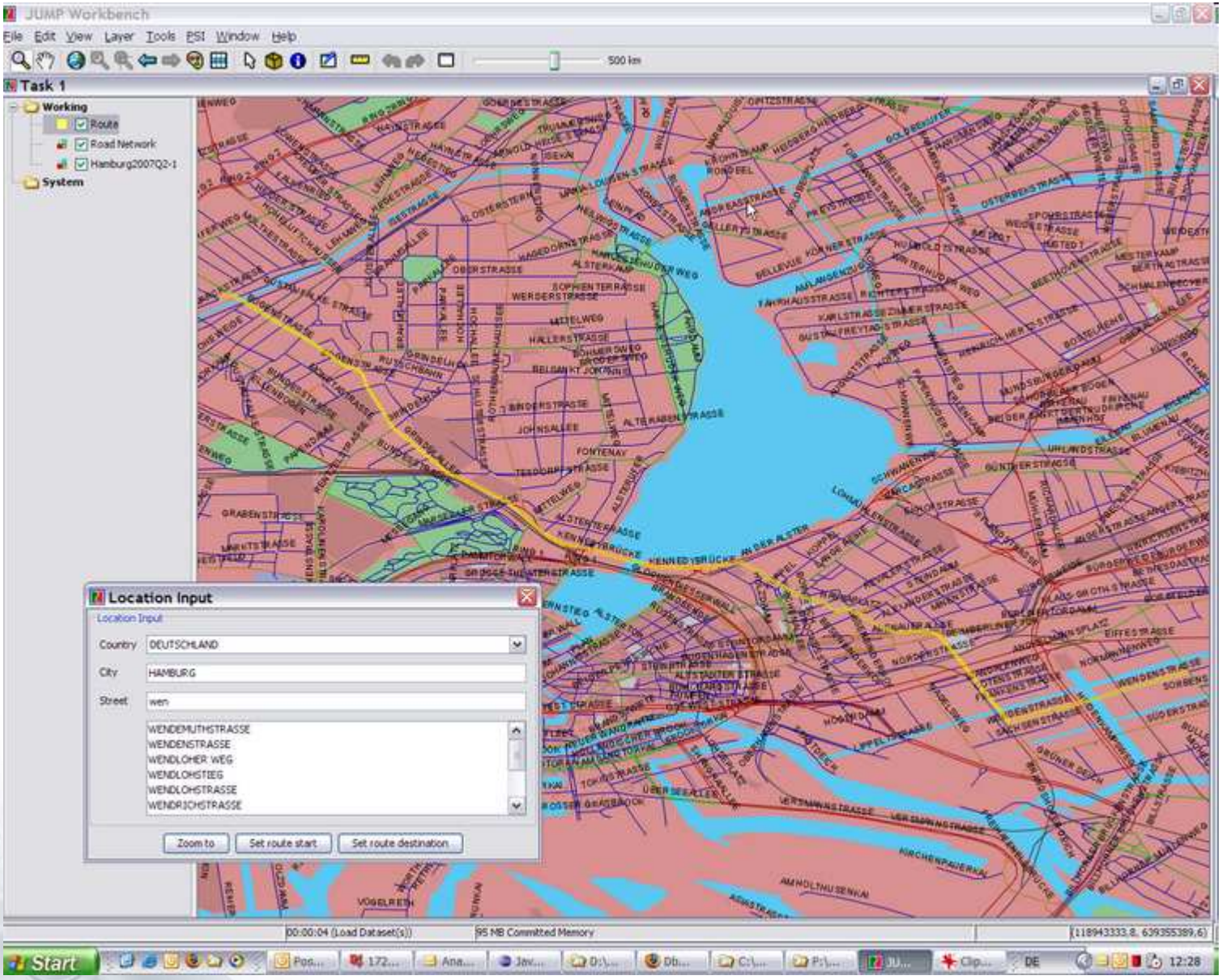
- **JUMP** (Java Unified Mapping Platform)
- As yet, the main viewer for Anaconda products

- **uDig** is another open source platform for geodata
- Built on top of
  - Eclipse RCP
  - Geotools

- Will replace JUMP for Anaconda.

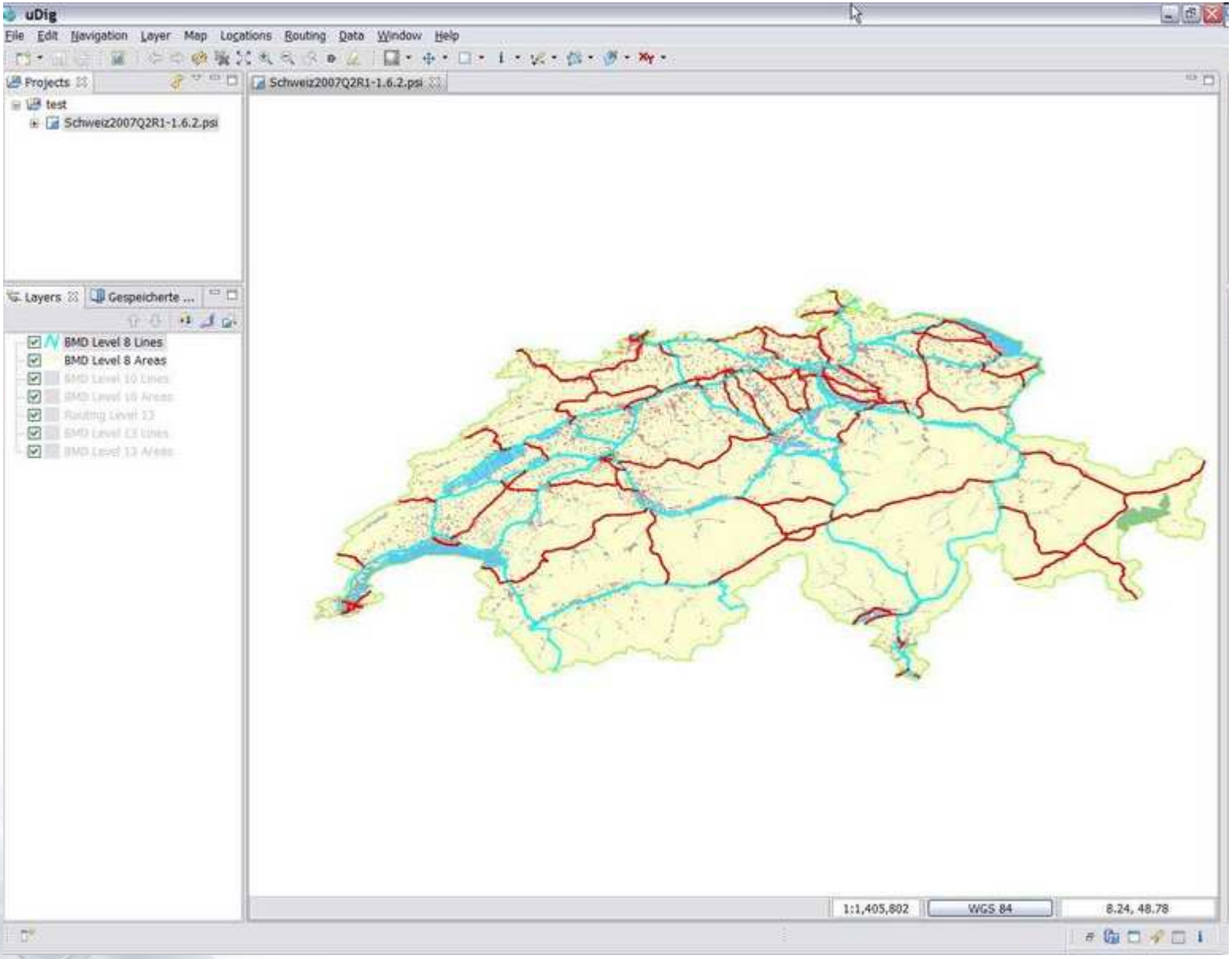
JUMP Screenshot

Harman International



uDig Screenshot

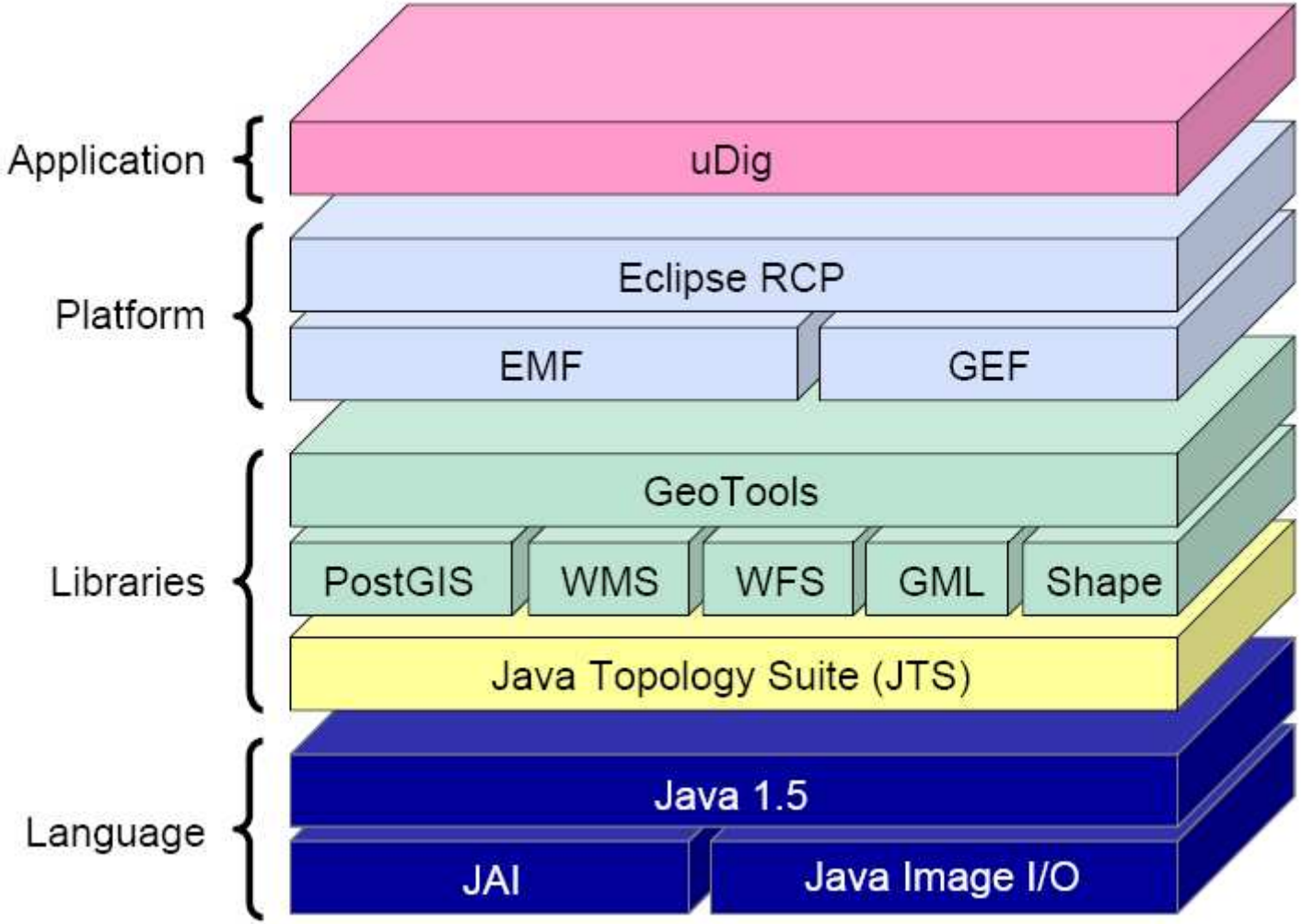
Harman International



uDig Benefits

- Based on OSGi and Eclipse RCP
- uDig plugins use standard Eclipse technology
- Plugin API for custom renderers (e.g. OpenGL, Java3D)
- Support for coordinate transformations
- LGPL license (vs. GPL for JUMP)
- Active and responsive community

uDig Architecture



Source: http://udig.refractions.net/files/docs/osg05TechnologySession.pdf

uDig Downsides

- Steep learning curve

- Third Party Hell
  - A mega-bundle `net.refractions.udig.libs` contains Geotools and all external dependencies.
  - We are working with the communities to solve this.

Features of our Viewer Plug-ins

- DataStore implementation for the NDS database format
- Multiple levels of detail in 2D map
- Scale-dependent layer selection
- Location input via Name Browser with automatic speller
- Route Calculation

Further Plans

uDig shall be the basis of an integrated Anaconda Workbench

- Map Viewer
- Database Compilation Console
- Quality Assurance

Links

- DataScript
  - http://datascript.berlios.de
- uDig
  - http://udig.refractions.net
- Geotools
  - http://www.geotools.org

- My Blog
  - http://hwellmann.blogspot.com

