

JGRASS – present and Future

HydroloGIS
Andrea Antonello
Silvia Franceschi

University of Trento
Prof. Riccardo Rigon

Foss4G2009 - Sydney 23 October 2009

The road to JGrass

2002

Horton Machine in GRASS

The road to JGrass

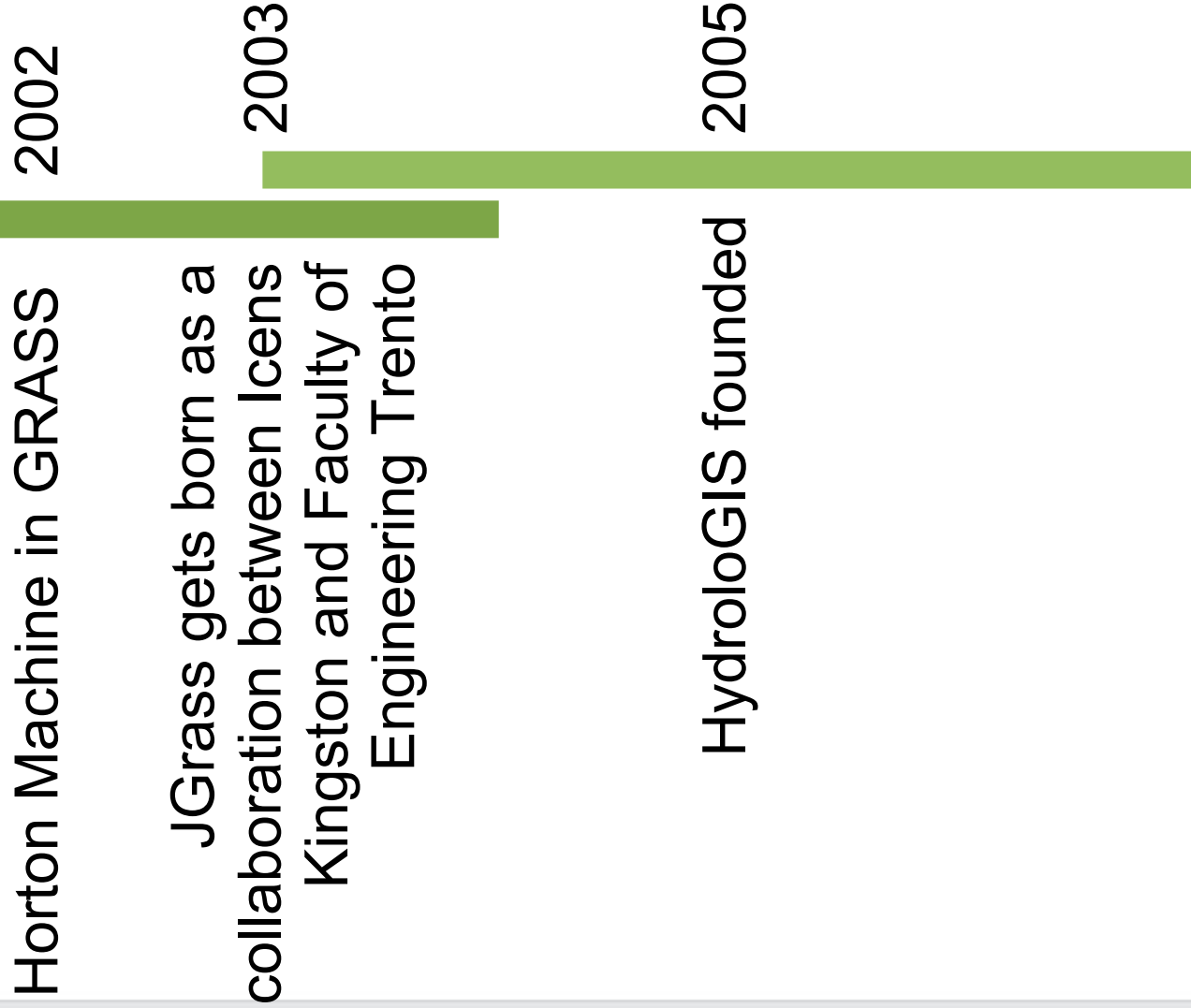
2002

Horton Machine in GRASS

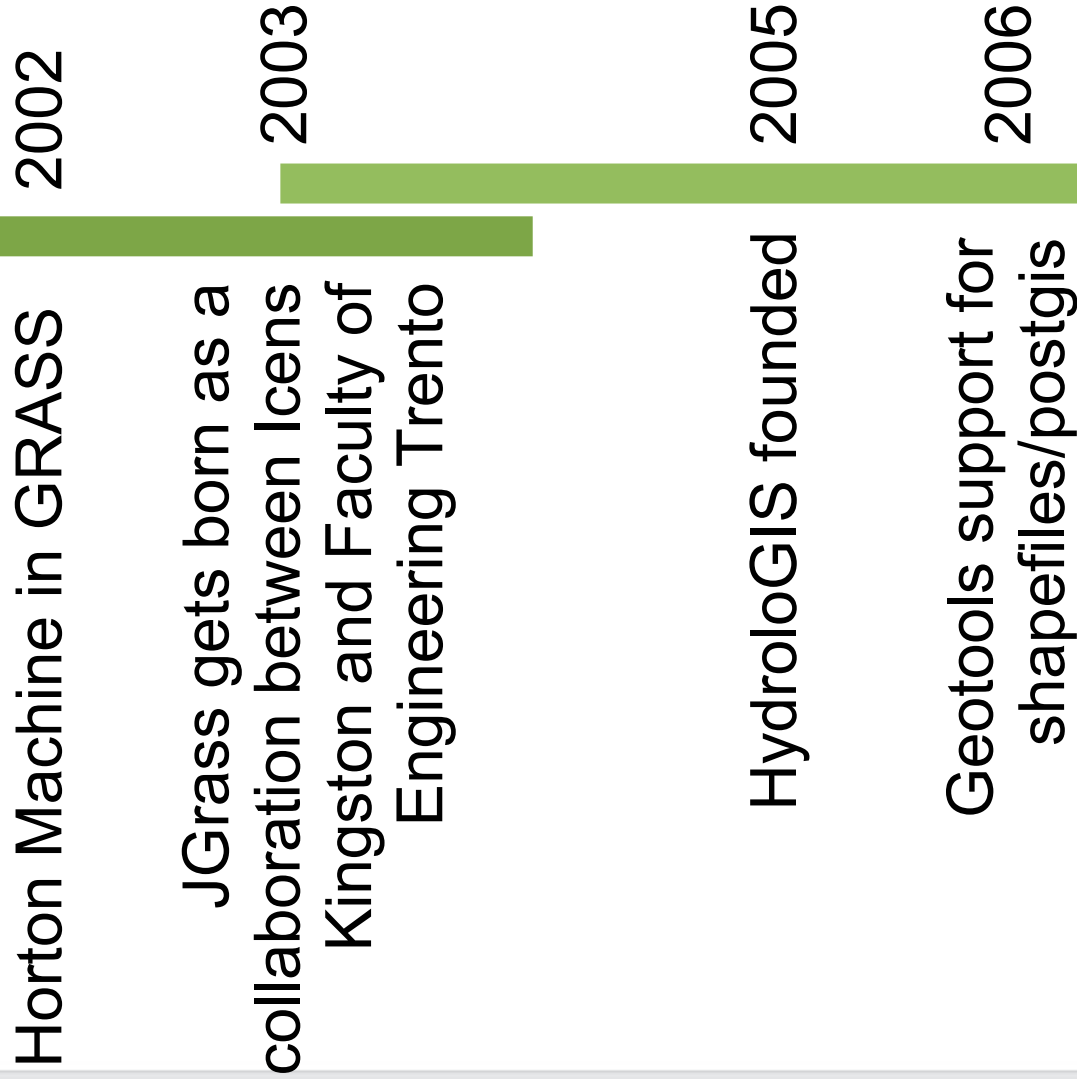
2003

JGrass gets born as a
collaboration between Icnens
Kingston and Faculty of
Engineering Trento

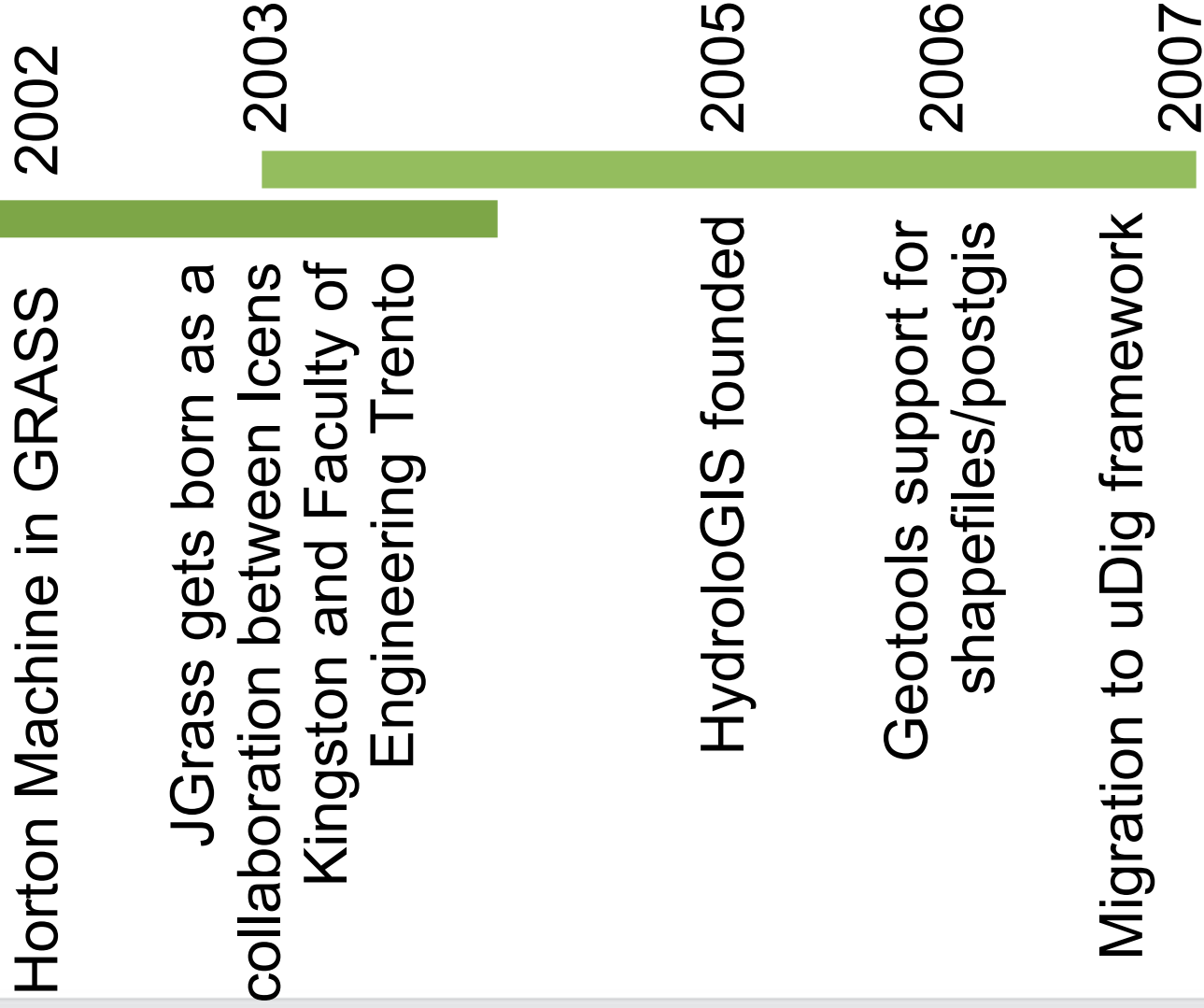
The road to JGrass



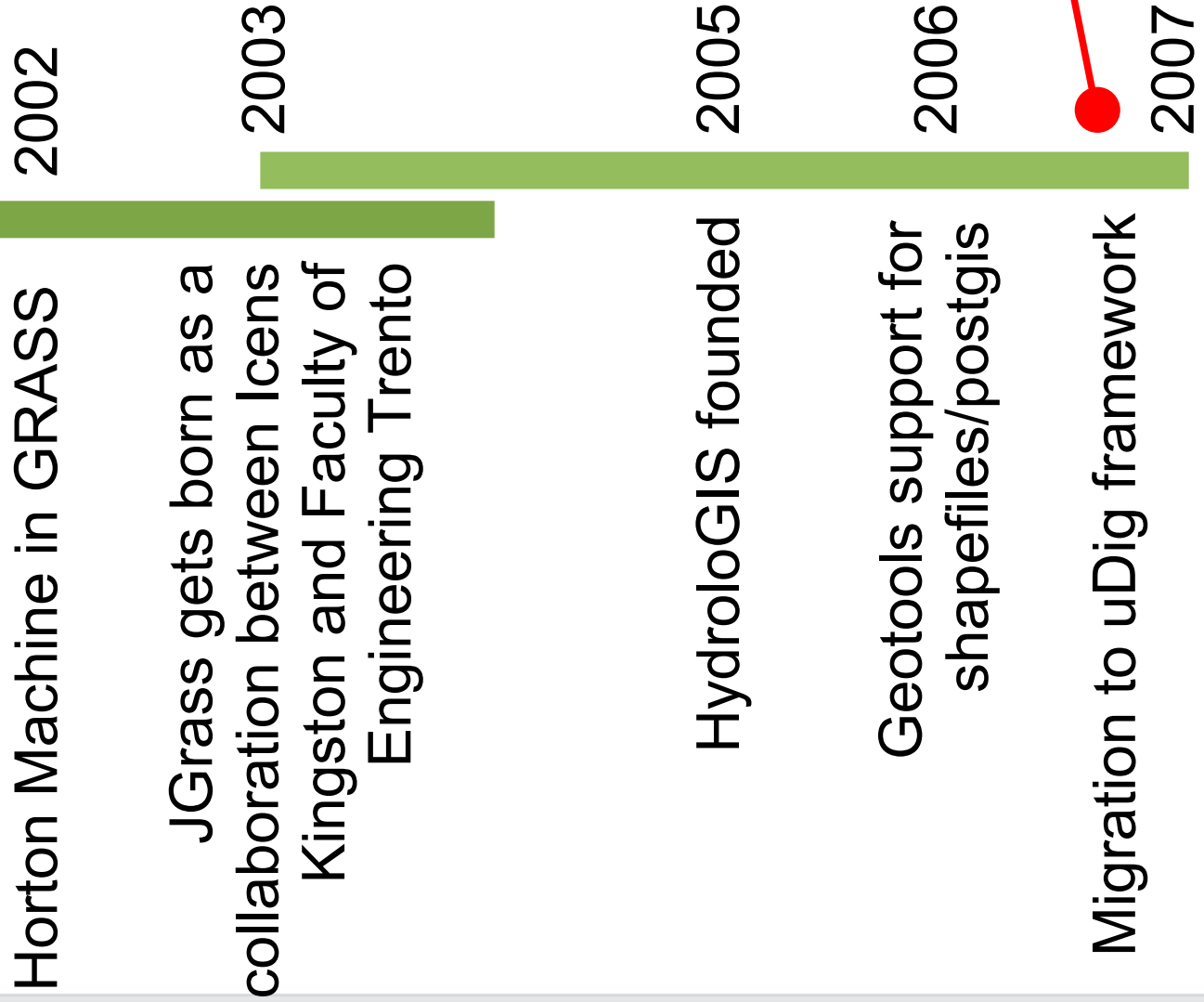
The road to JGrass



The road to JGrass



The road to JGrass



Java GPL

Horton Machine in GRASS

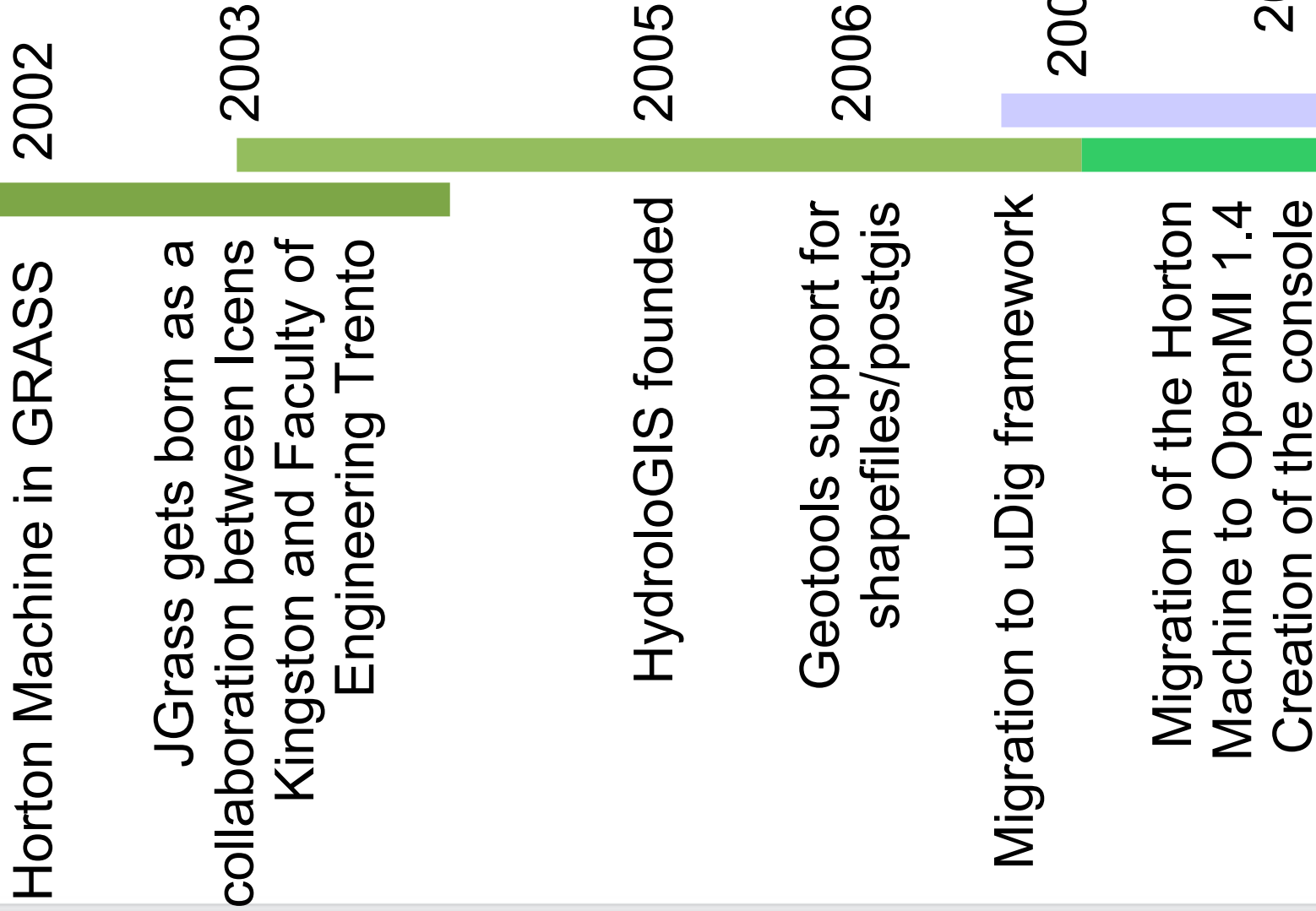
JGrass gets born as a collaboration between Icnens Kingston and Faculty of Engineering Trento

HydroloGIS founded

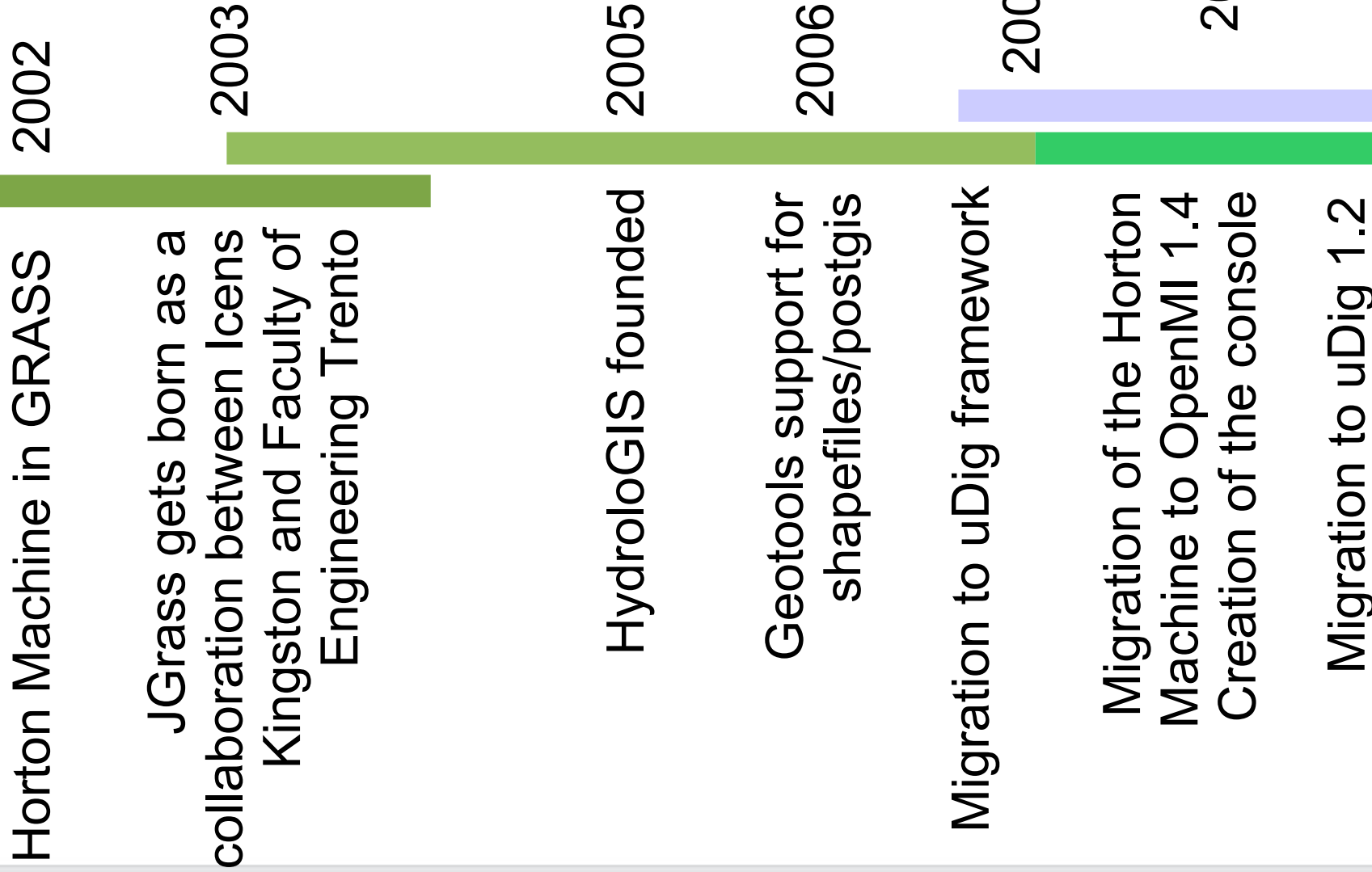
Geotools support for shapefiles/postgis

Migration to uDig framework

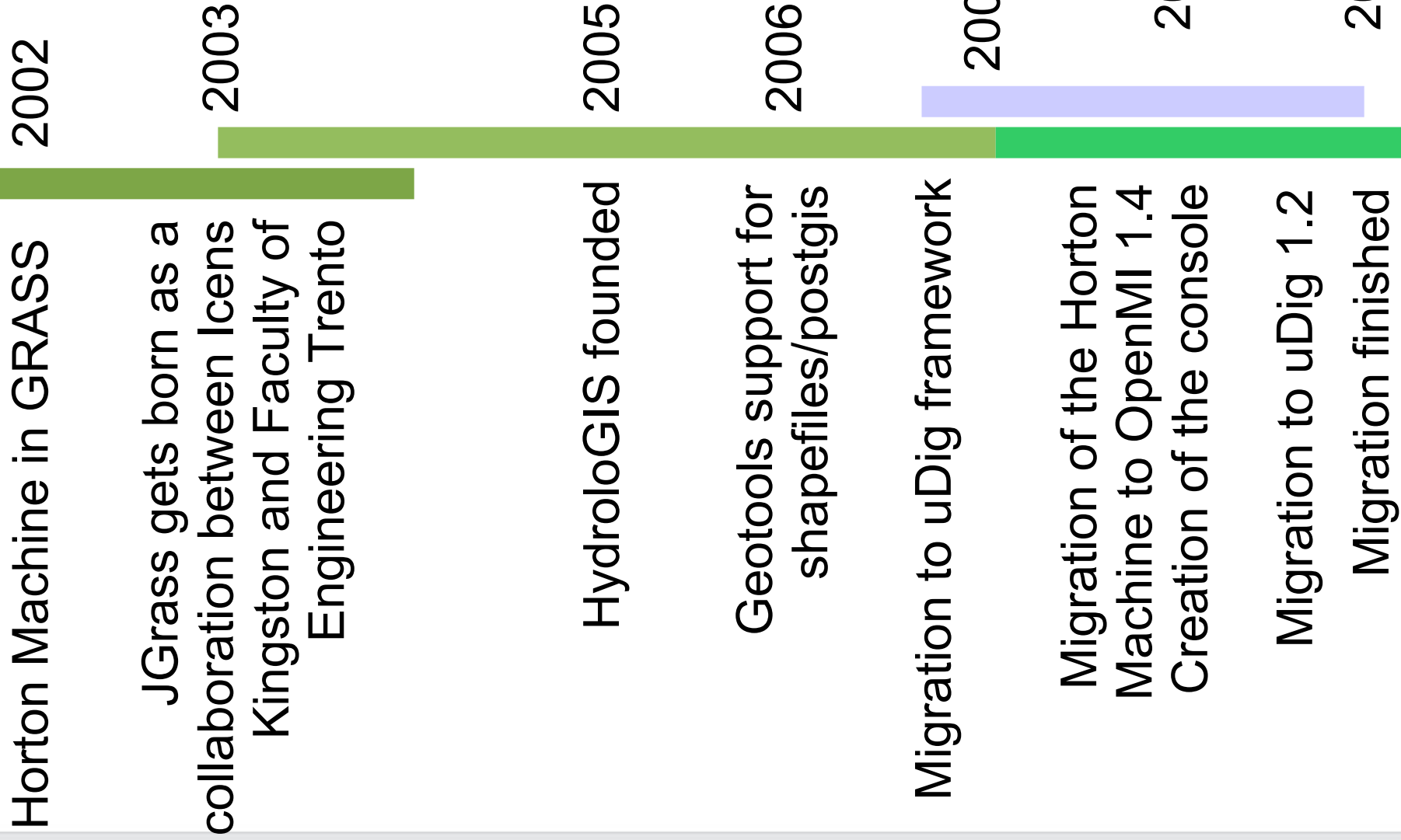
The road to JGrass



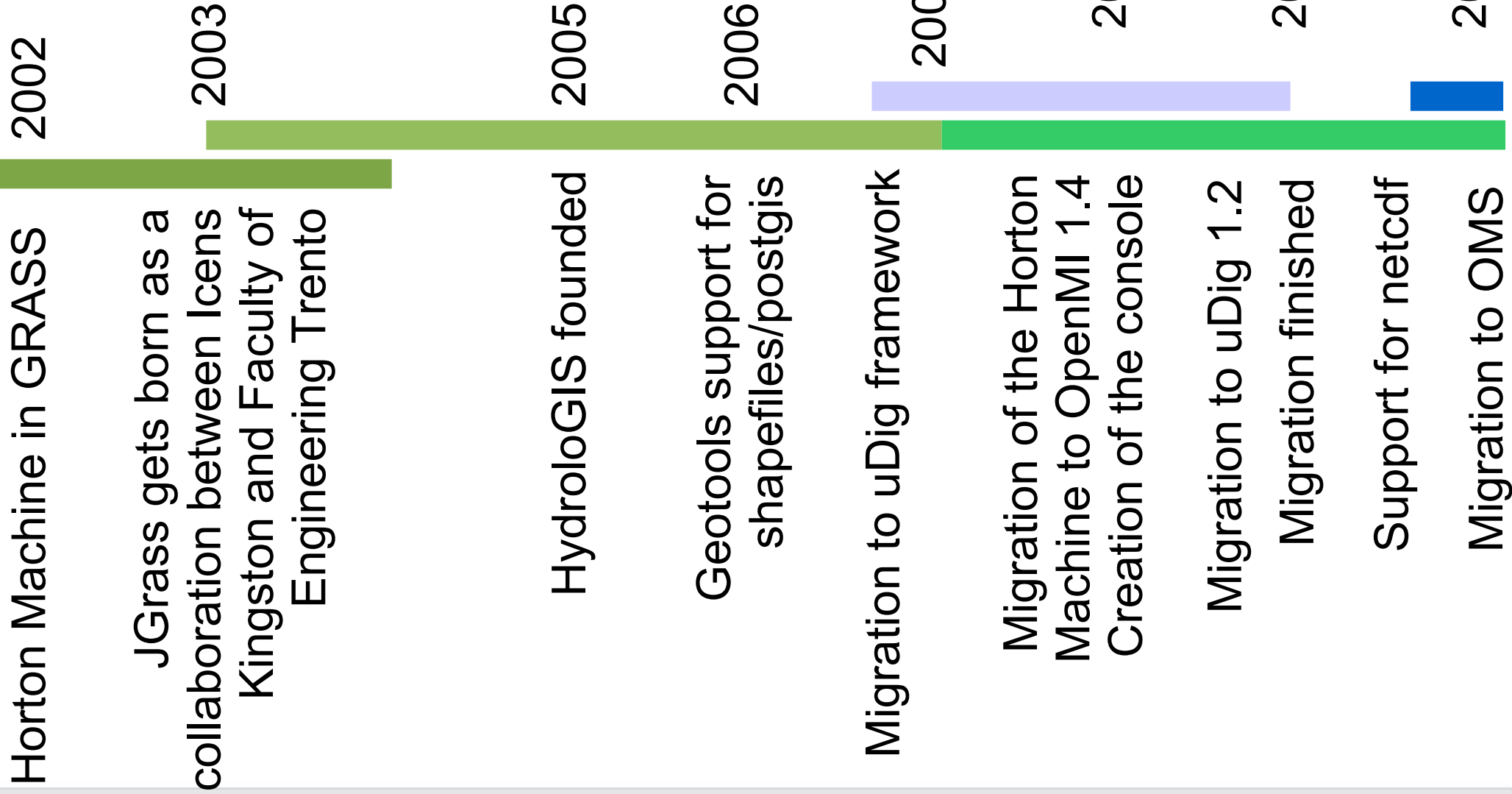
The road to JGrass



The road to JGrass



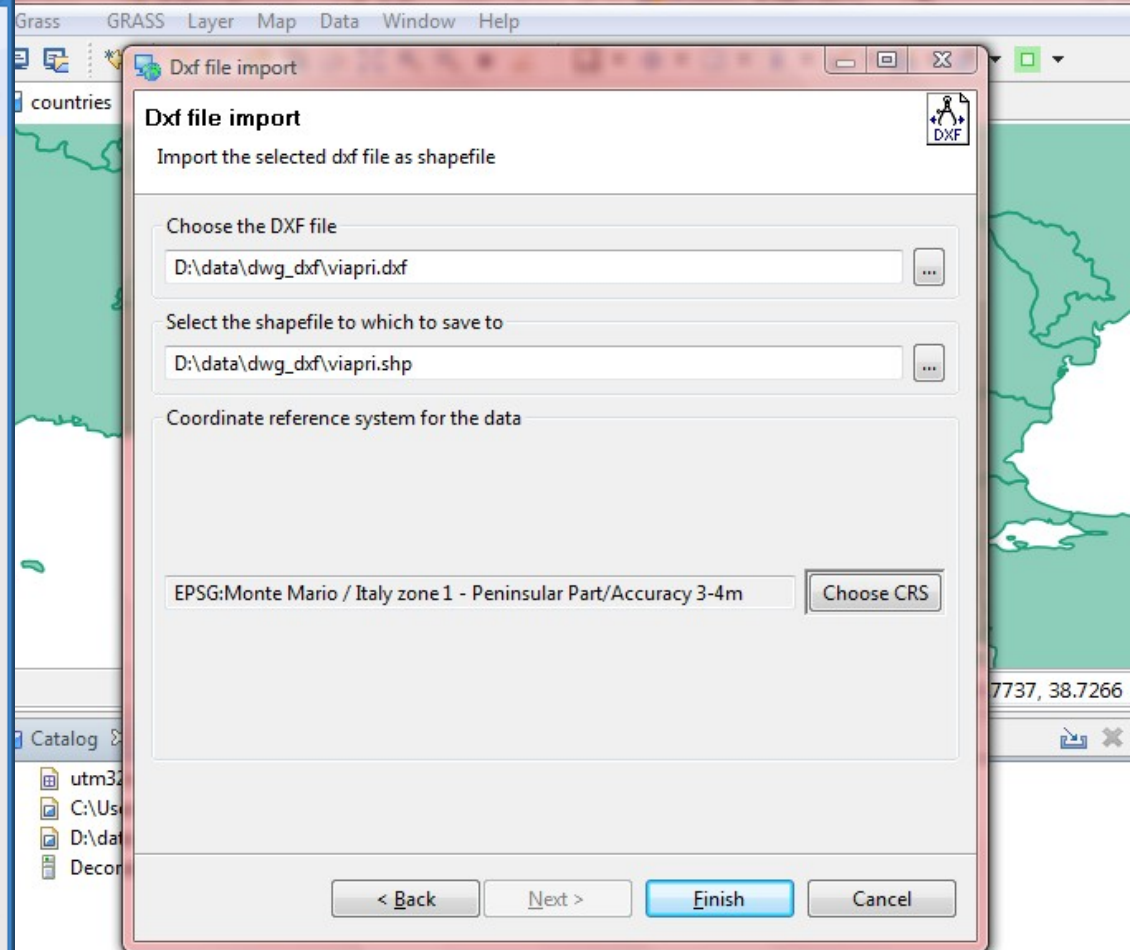
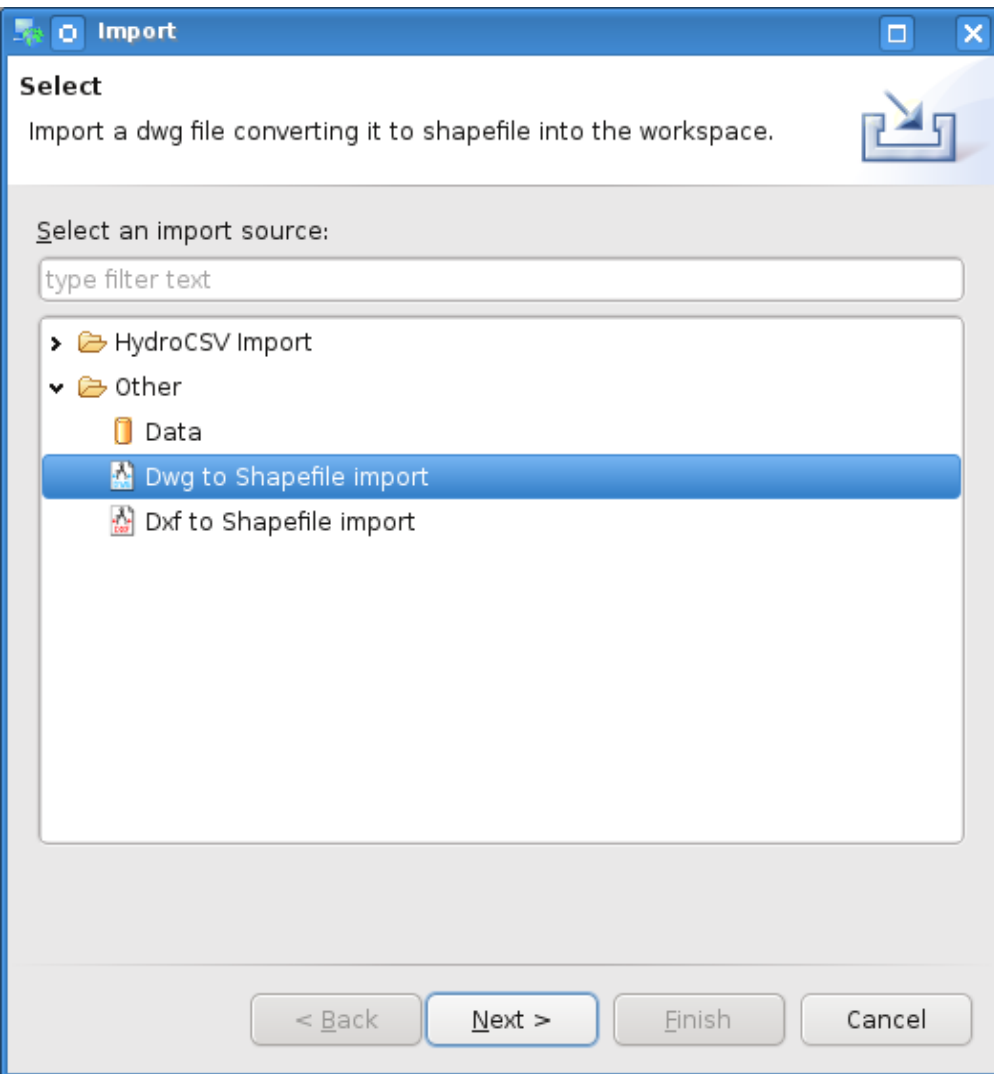
The road to JGrass



What have we been working on lately?

JGrass – handling DXF and DWG

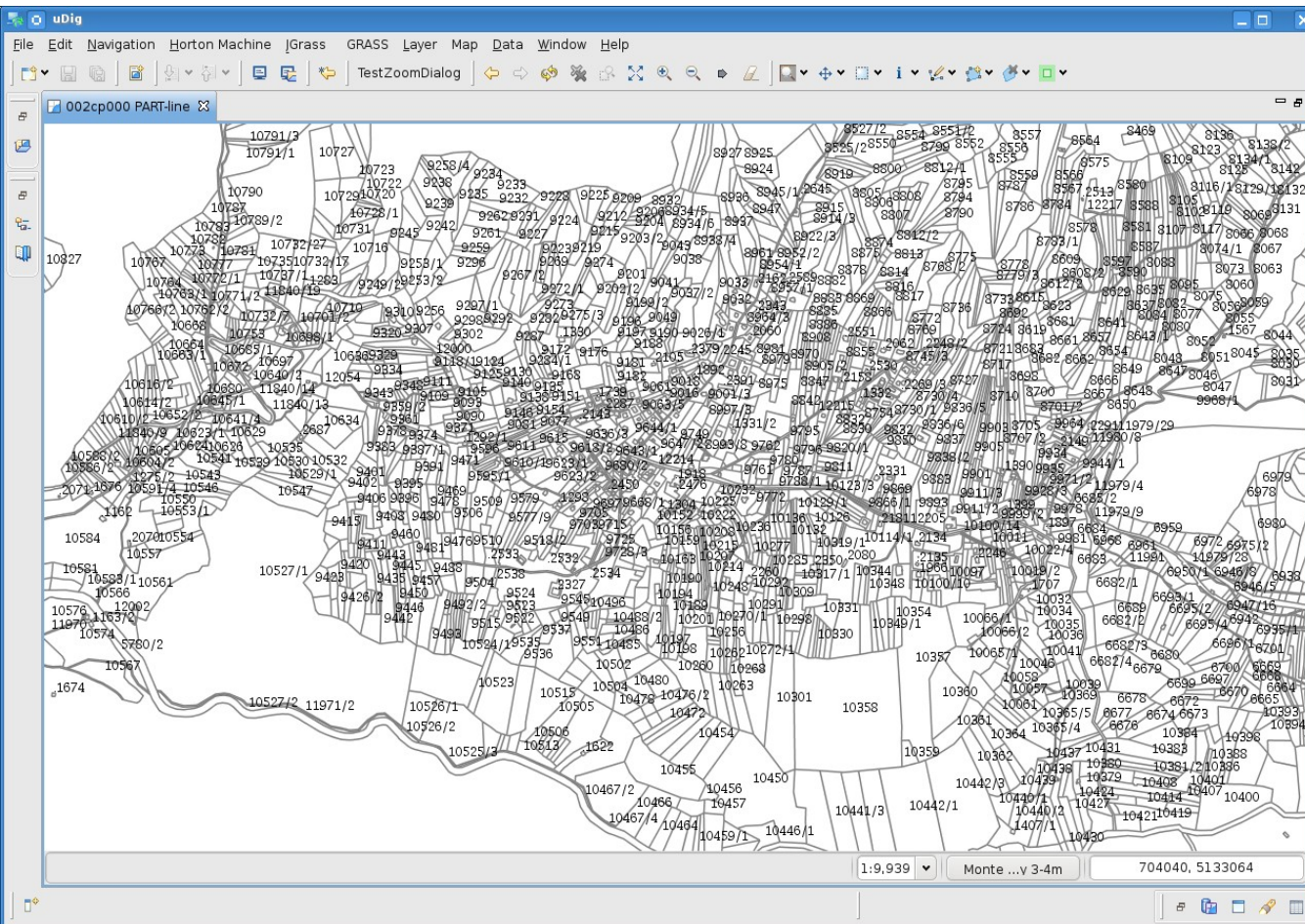
Support for DXF and DWG (up to 2000) files (are file formats used for storing design data and metadata)



JGrass – handling DXF and DWG

How are DXF files imported?

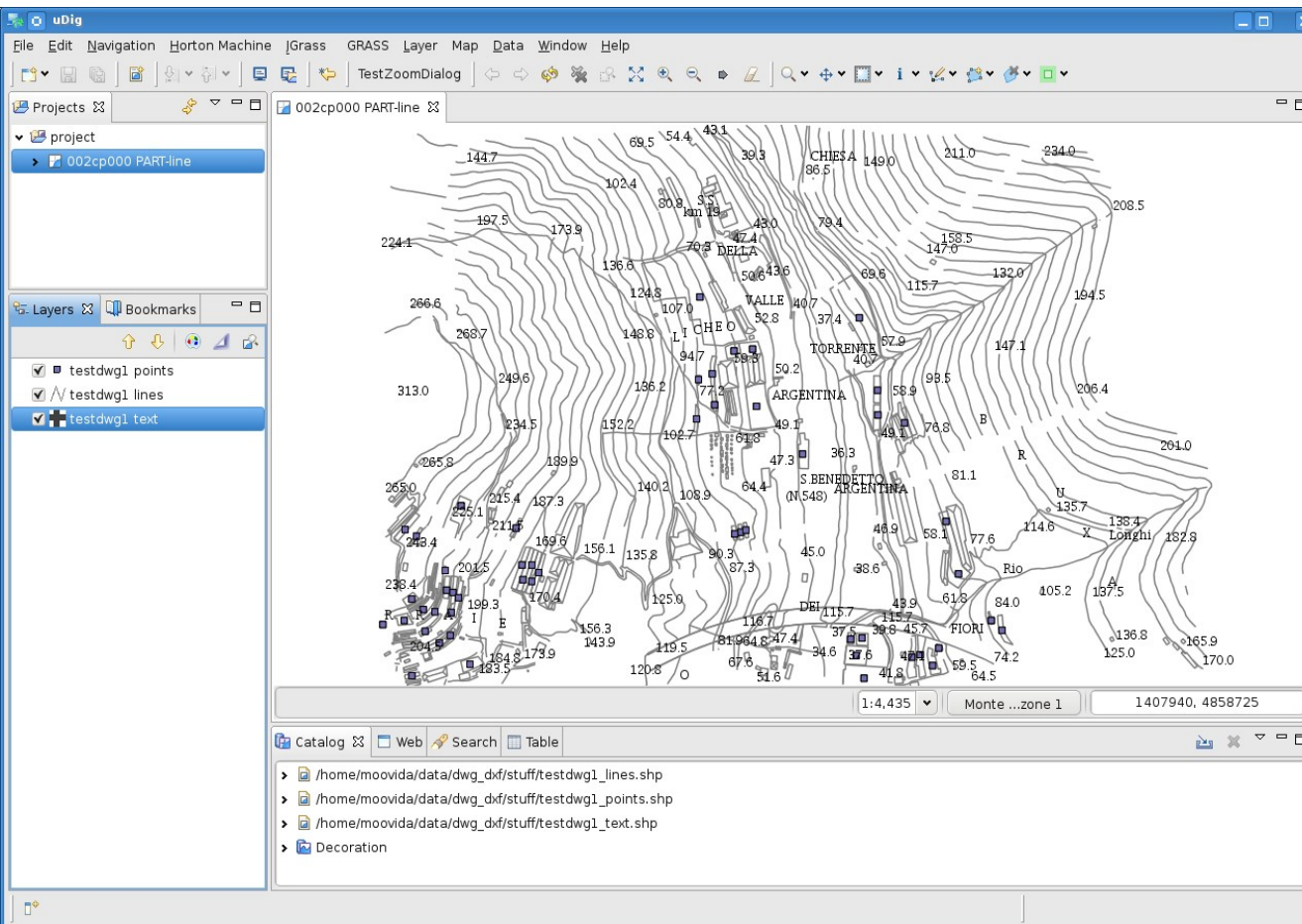
- every layer is imported as an own shapefile of the type contained
- the types that can be represented through points, lines and polygons, are imported, the others ignored
- text is imported as point layer with a text field to be used as label



JGrass – handling DXF and DWG

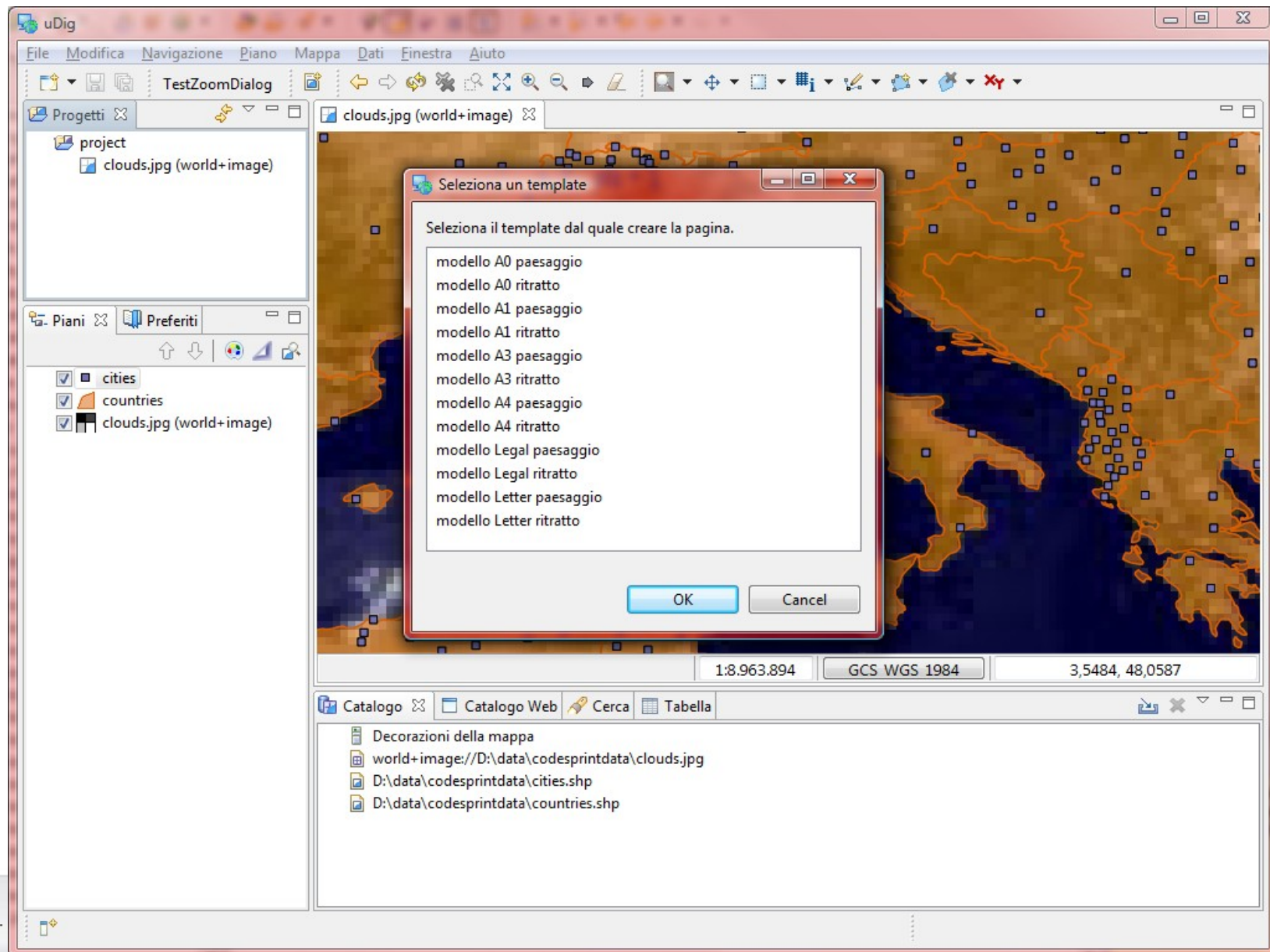
How are DWG files imported?

- given the complexity and due to the library, all the data is imported in 3 main shapefiles of the types: points, lines and polygon
- the original layer name is put as an attribute in order to be able to select data from one layer and copy/paste it to a new layer if needed



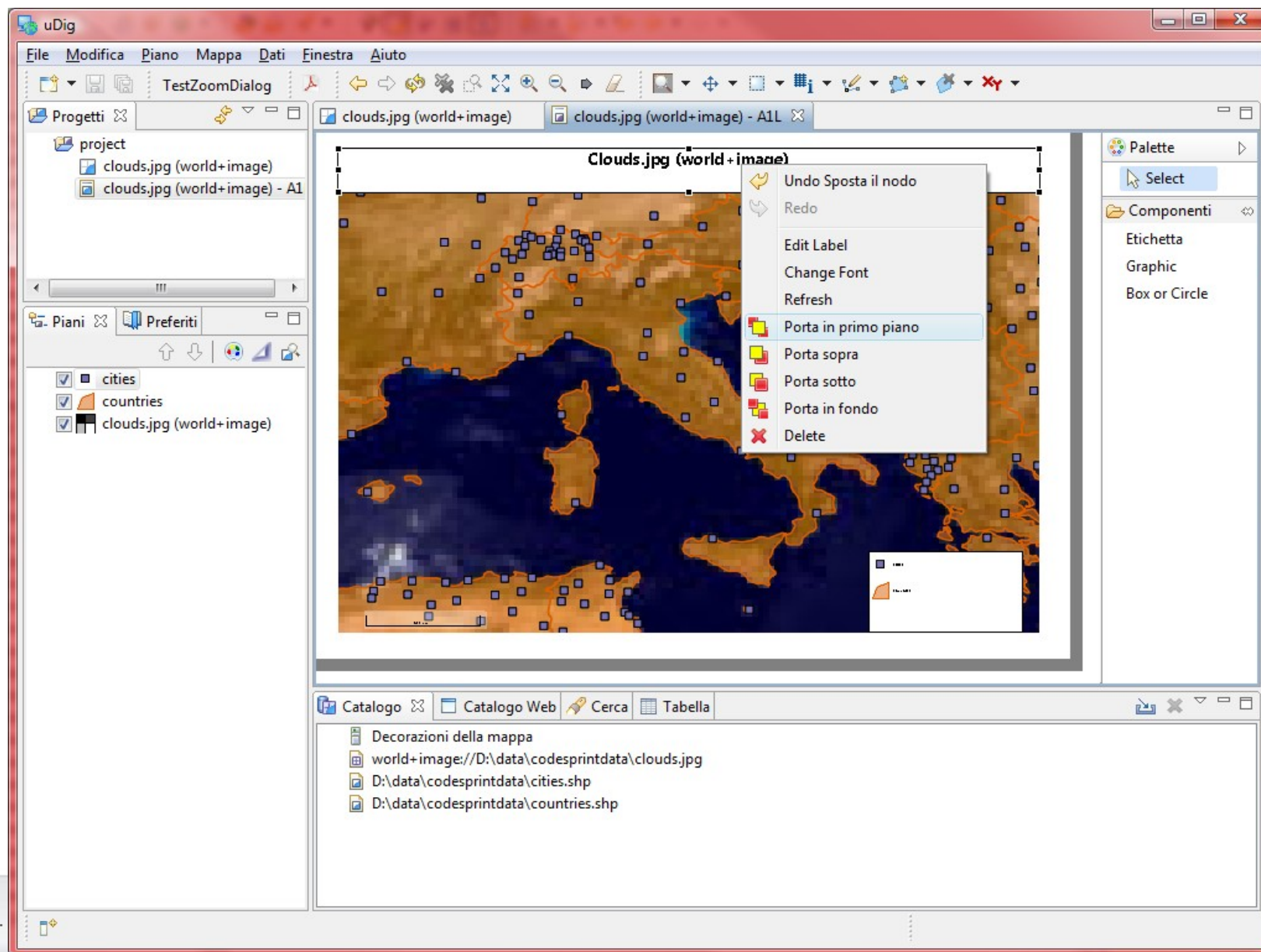
JGrass – enhancements of the printing engine

Choose your page template...



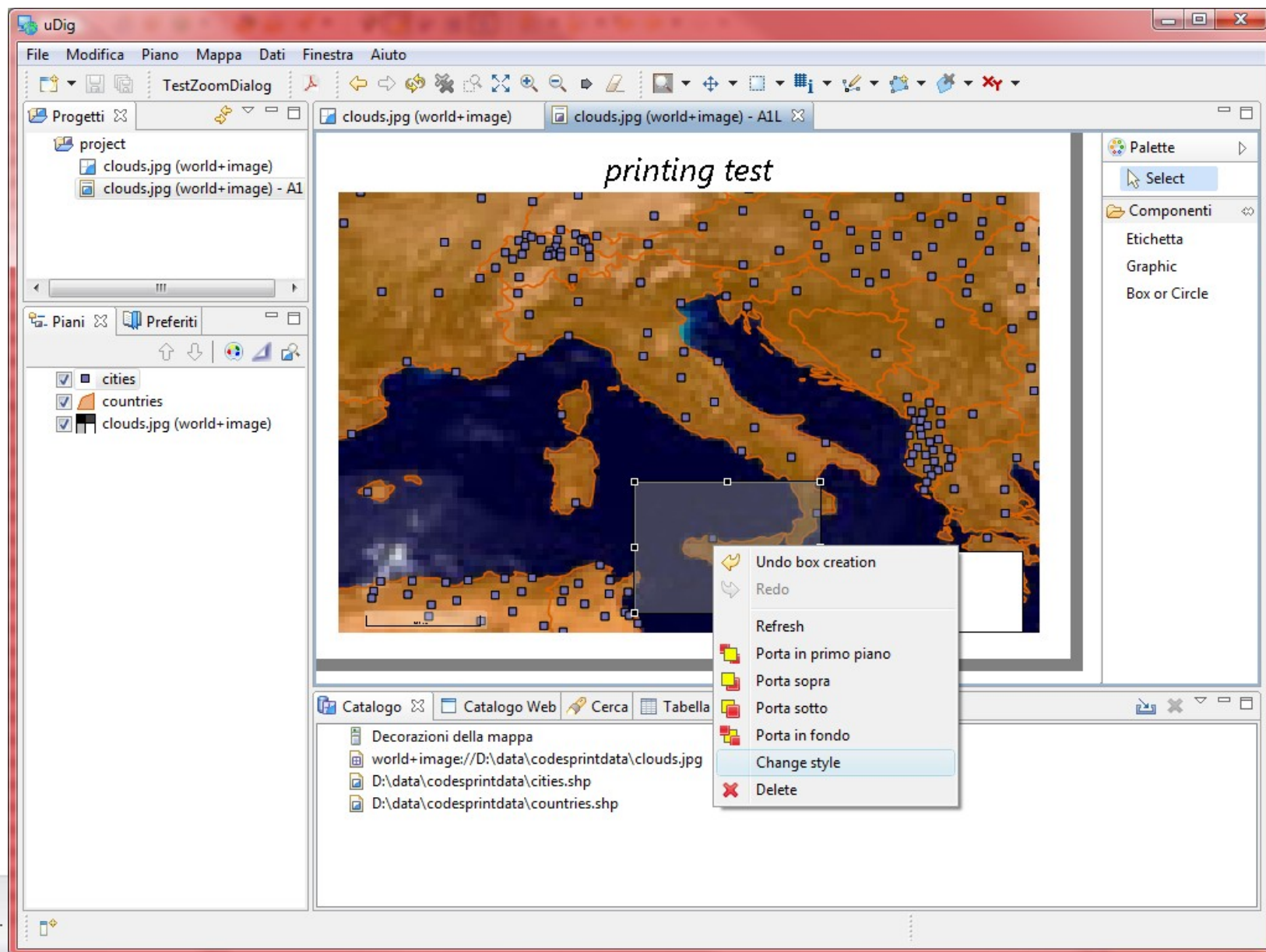
JGrass – enhancements of the printing engine

...modify the template, adapt it...



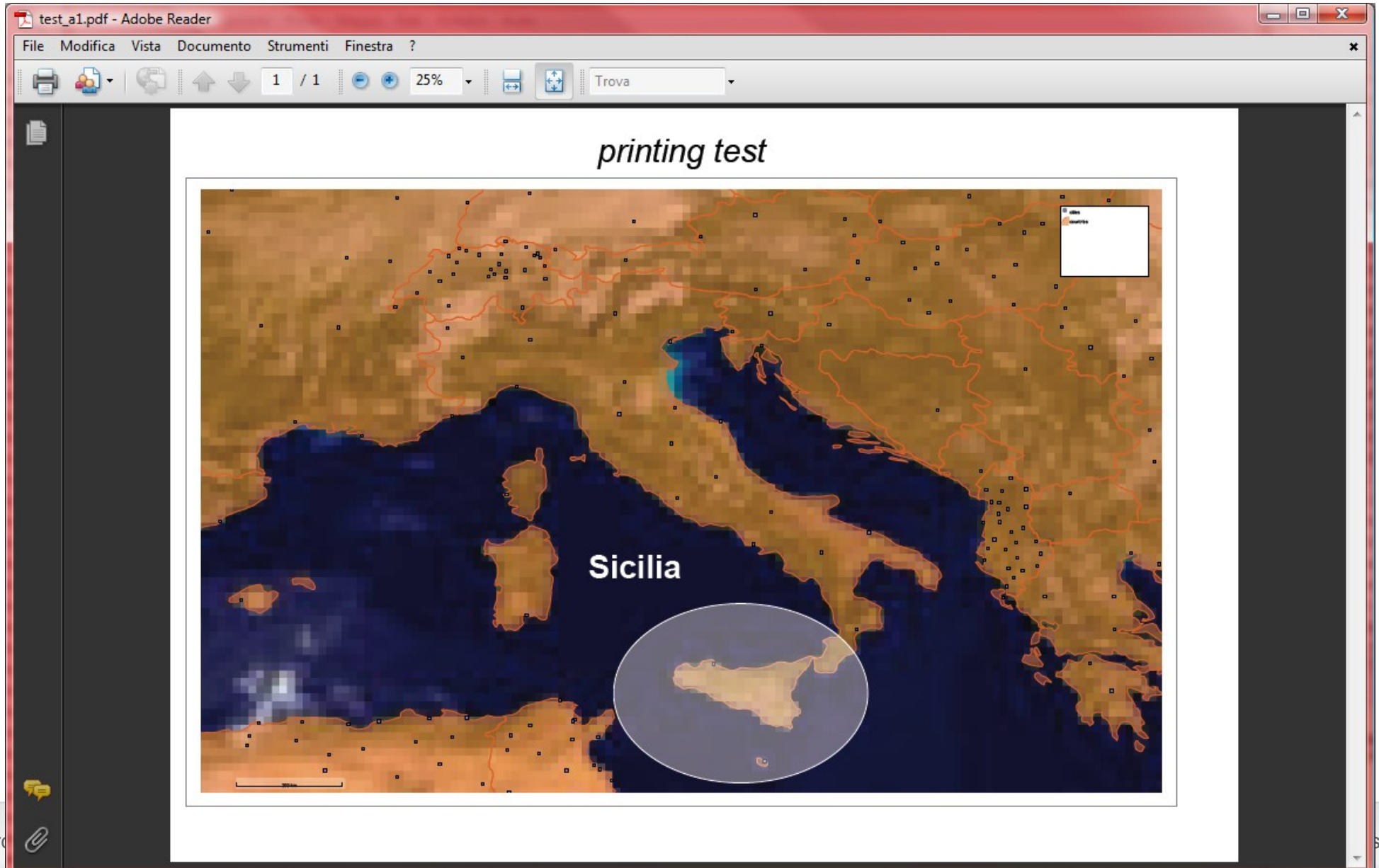
JGrass – enhancements of the printing engine

...add shapes or graphics...



JGrass – enhancements of the printing engine

...and finally print it to pdf.



JGrass – handling netcdf files

NetCDF (Network Common Data Form) is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.

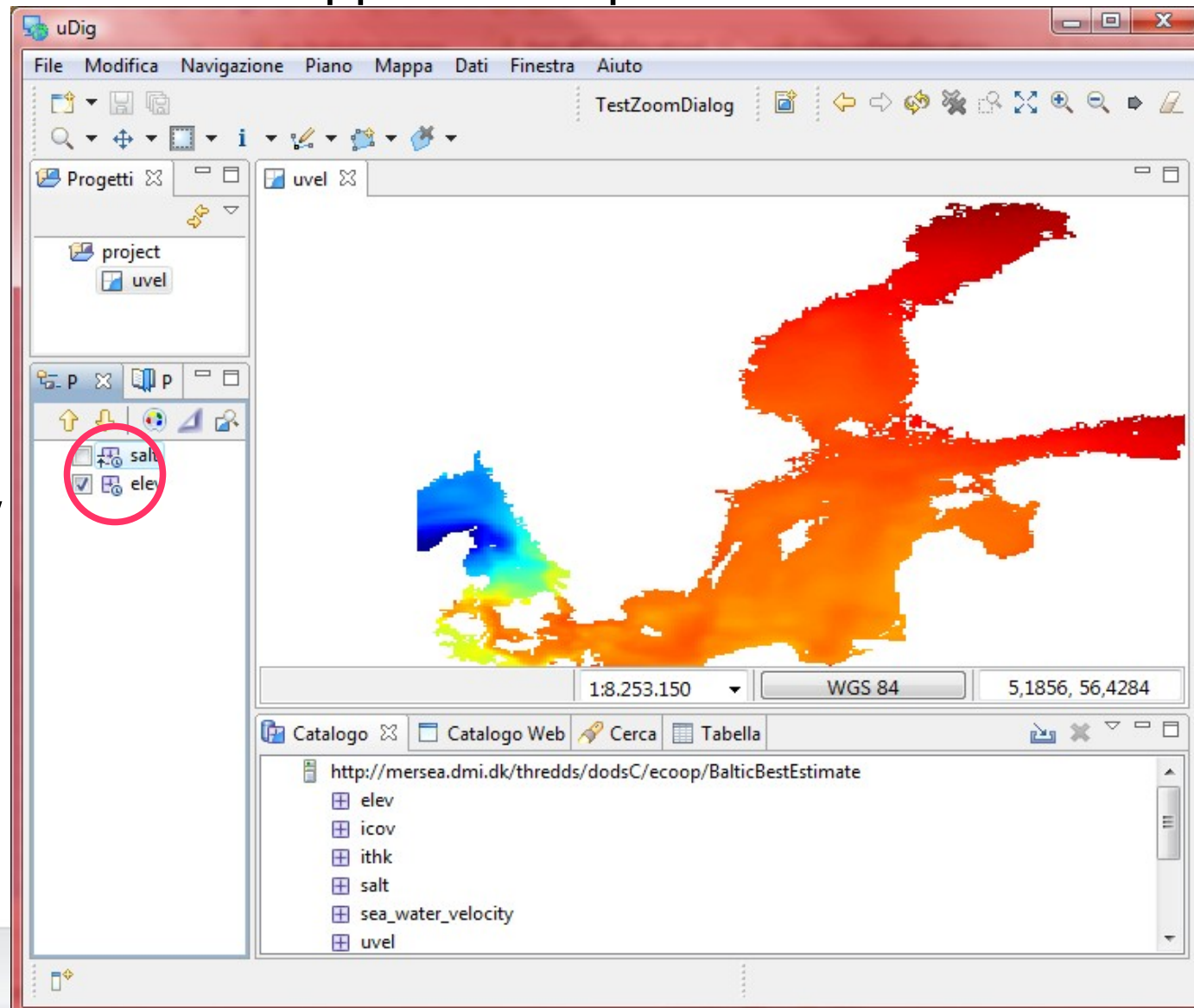
The project is primarily driven by the Unidata program at the University Corporation for Atmospheric Research (UCAR). They are also the chief source of netCDF software, standards development, updates etc. The format is an open standard.

Netcdf support was implemented with the fundings from the Google Summer of Code 2009 Program.

JGrass – reading netcdf files

NetCDF is supported for map visualization up to 4D. 2D is handled as a normal raster map, 3D and 4D add support for depth and time levels.

- both the remote (opendap) and local dataset handling is supported
- icons of the layer show the dimension of the dataset



JGrass – reading netcdf files

The depth and time properties, if available, can be browsed

The screenshot displays the uDig GIS application window. The main map area shows a coastal region with a blue landmass and a yellow/green coastal area. A dialog box titled "Set new property" is open, allowing the user to select a timestep and a vertical value. The vertical value dropdown is open, showing options: -4.0, -9.0, -51.0, -96.0, and -175.0. The interface includes a menu bar (File, Modifica, Navigazione, Piano, Mappa, Dati, Finestra, Aiuto), a toolbar, a project browser on the left, and a catalog at the bottom.

uDig

File Modifica Navigazione Piano Mappa Dati Finestra Aiuto

TestZoomDialog

Progetti

project

uvel

Set new property

Select a timestep

Select a vertical value

OK Cancel

-4.0
-9.0
-51.0
-96.0
-175.0

1:8.253.150 WGS 84 5,2248, 55,3309

Catalogo Catalogo Web Cerca Tabella

<http://mersea.dmi.dk/thredds/dodsC/ecoop/BalticBestEstimate>

- elev
- icov
- ithk
- salt
- sea_water_velocity
- uvel

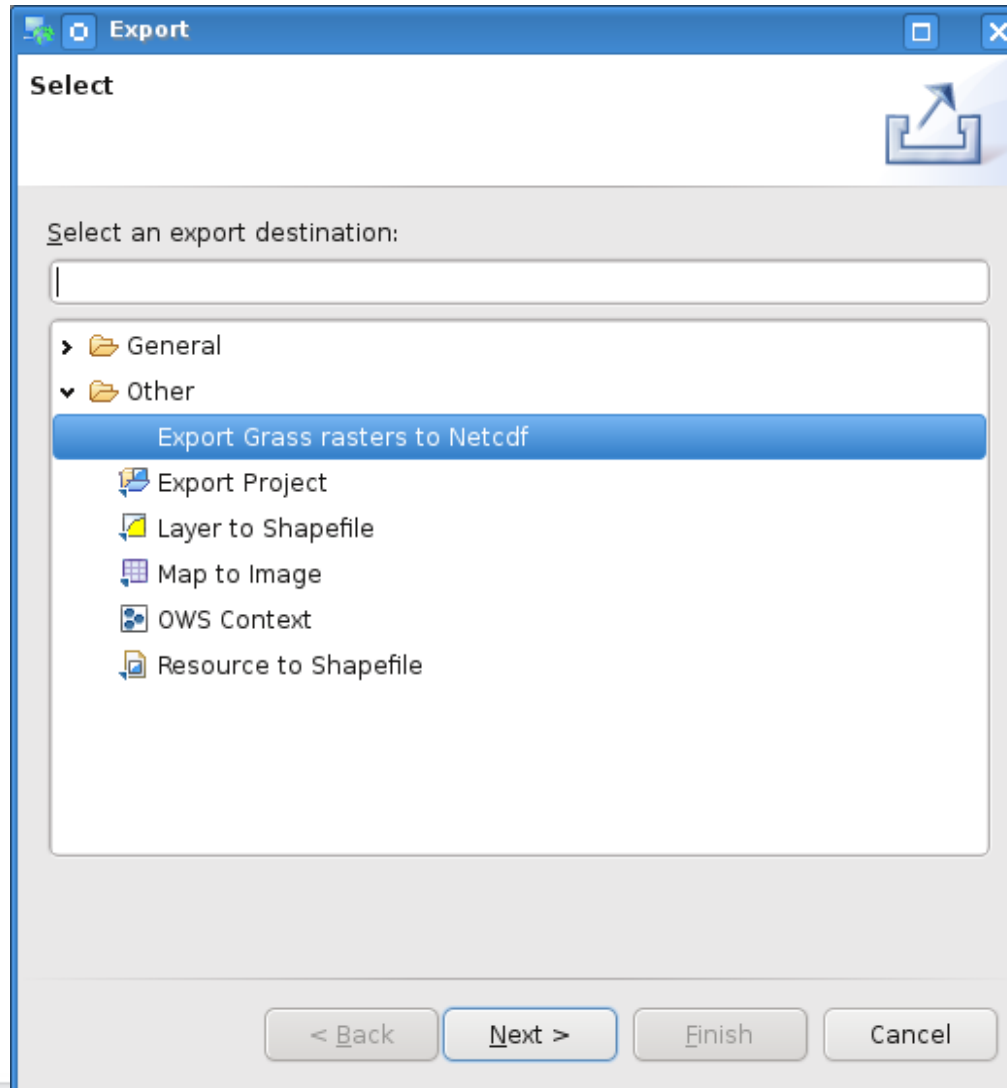
choose time and elevation of map

HydroloGIS s.r.l. - Via

www.hydrologis.com

JGrass – writing netcdf files

Raster maps from the workspace can be bundled and exported as NetCDF datasets.



JGrass – writing netcdf files

Step 1: define general metadata

Export Grass rasters to Netcdf

General parameters to define the exported netcdf dataset structure.

grass mapset to take maps from

output netcdf path

time [YYYY-MM-DD HH:MM]

First time moment

Last time moment

Time step in minutes

levels [m]

Comma separated list of levels

Global attributes

Conventions: CF-1.4

Contact: andrea.antonello@gmail.com

References: http://www.hydrologis.com

Comment: any comment here

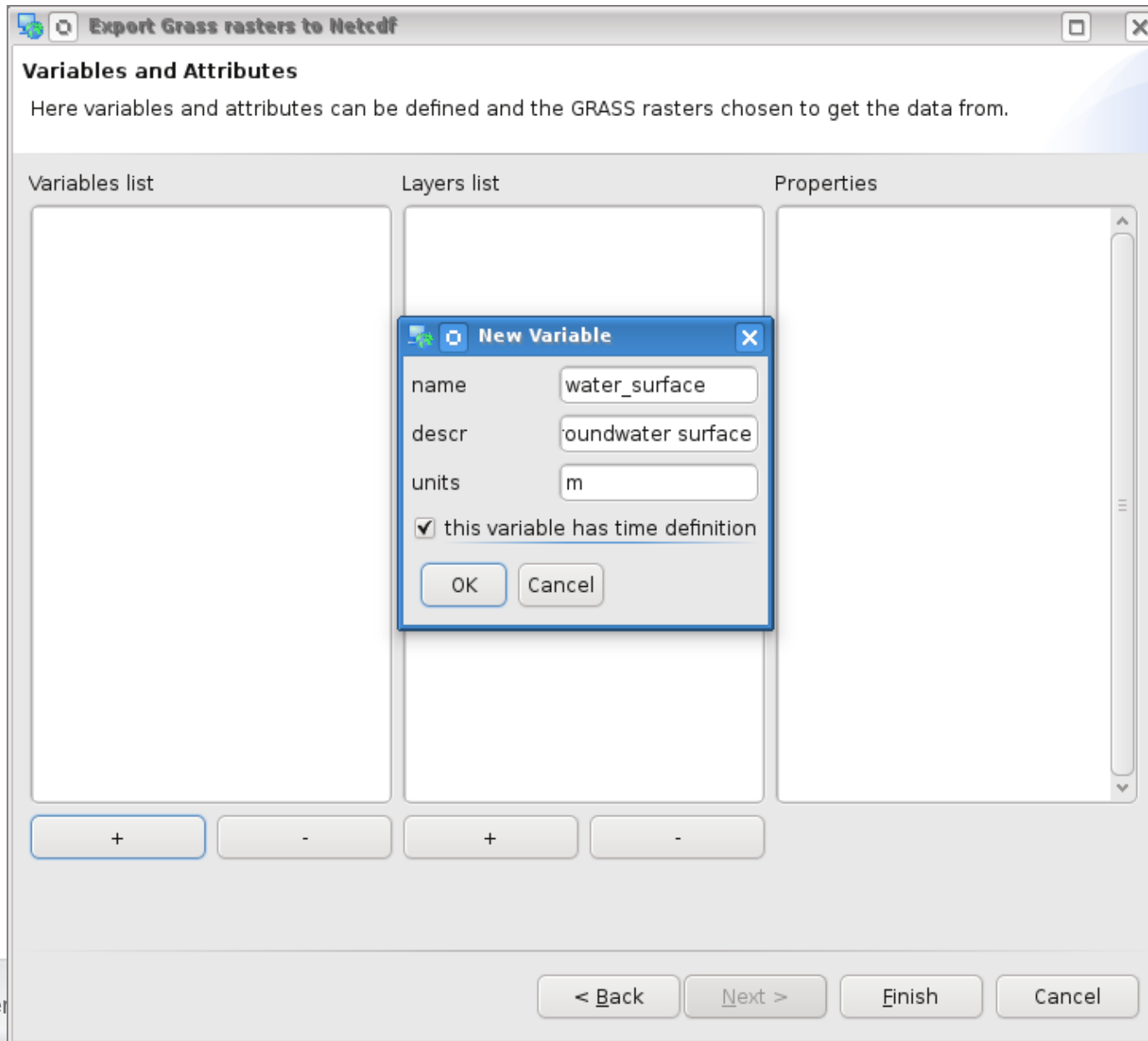
Distribution_statement: IN NO EVENT SHALL MY COMPANY OR ITS REPRESENTATIVES BE LIABLE... blah blah

+ ? -

< Back Next > Finish Cancel

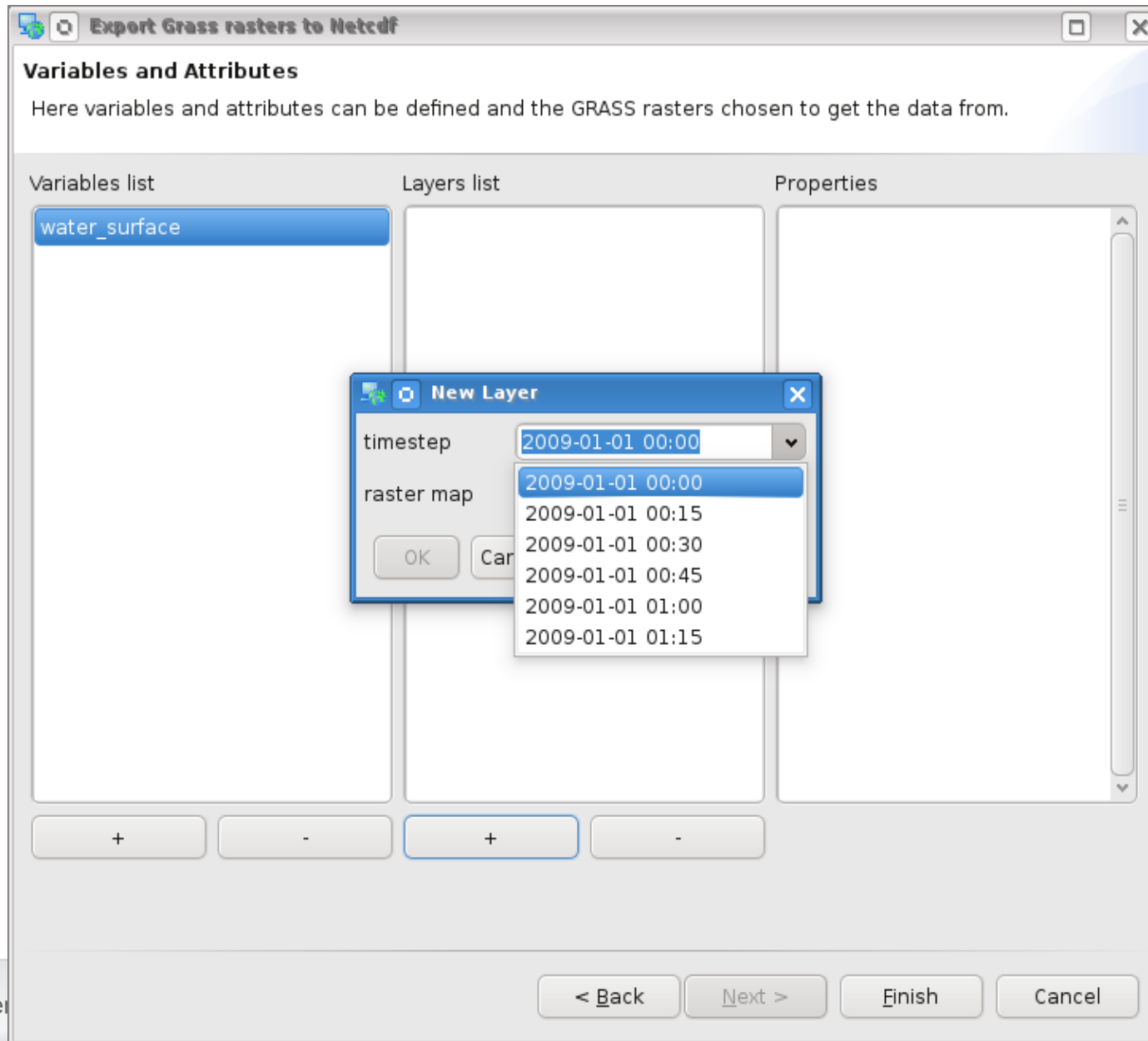
JGrass – writing netcdf files

Step 2: define variables, making sure time support is enabled if necessary



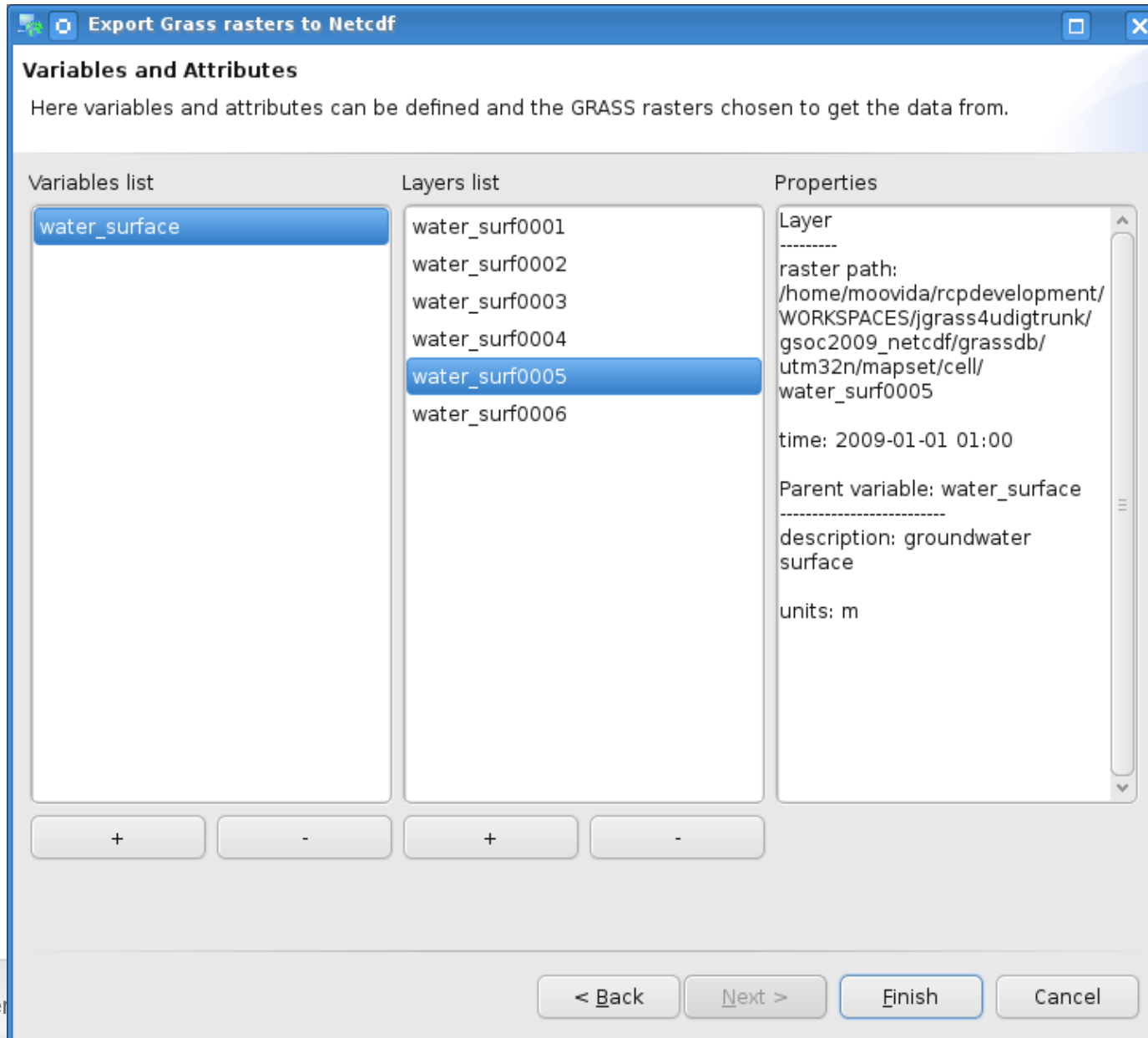
JGrass – writing netcdf files

Step 3: chose the raster maps to refer to particular timesteps, add them as layers



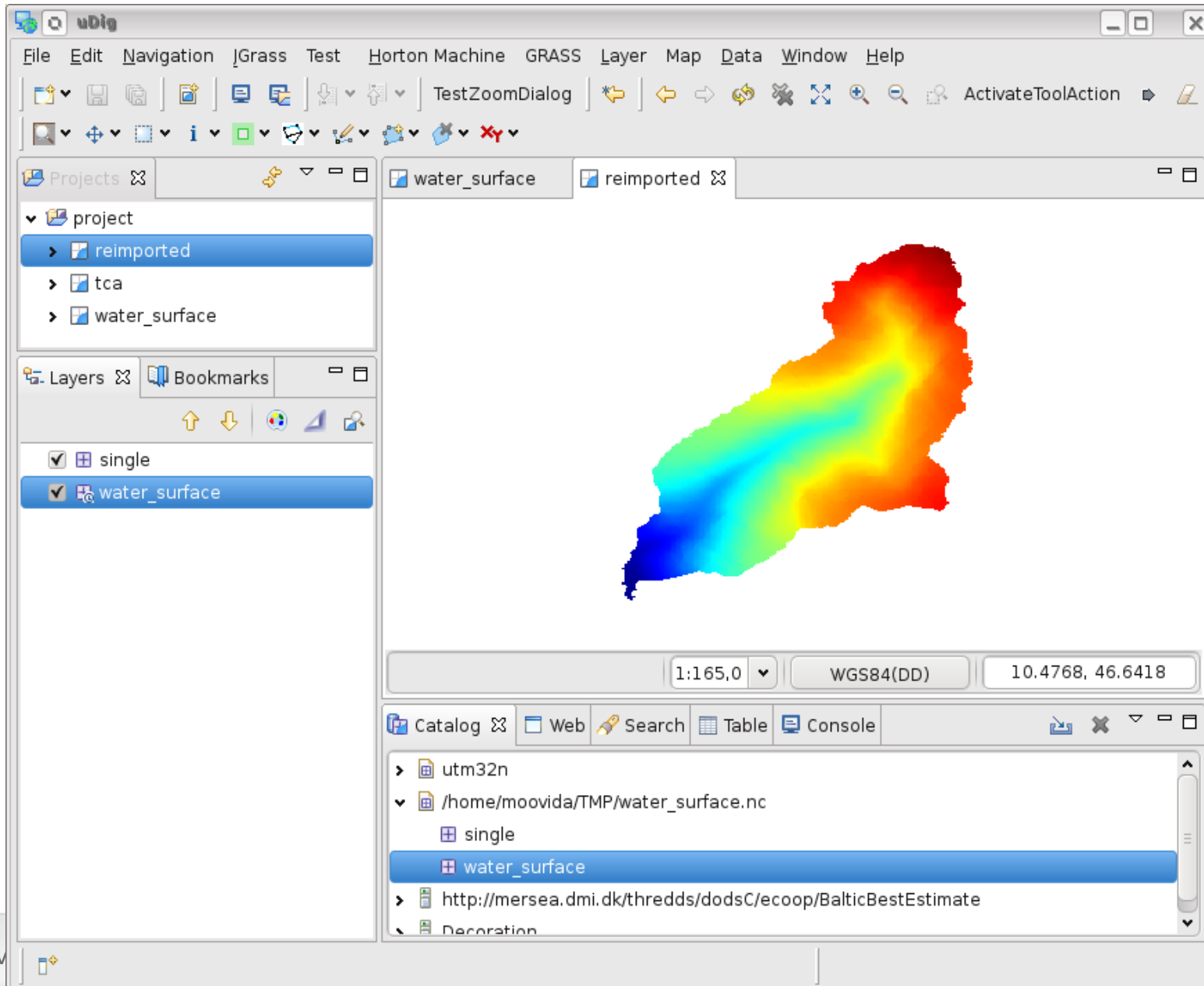
JGrass – writing netcdf files

Step 3: chose the raster maps to refer to particular timesteps, add them as layers



JGrass – writing netcdf files

Step 4: check the exported dataset



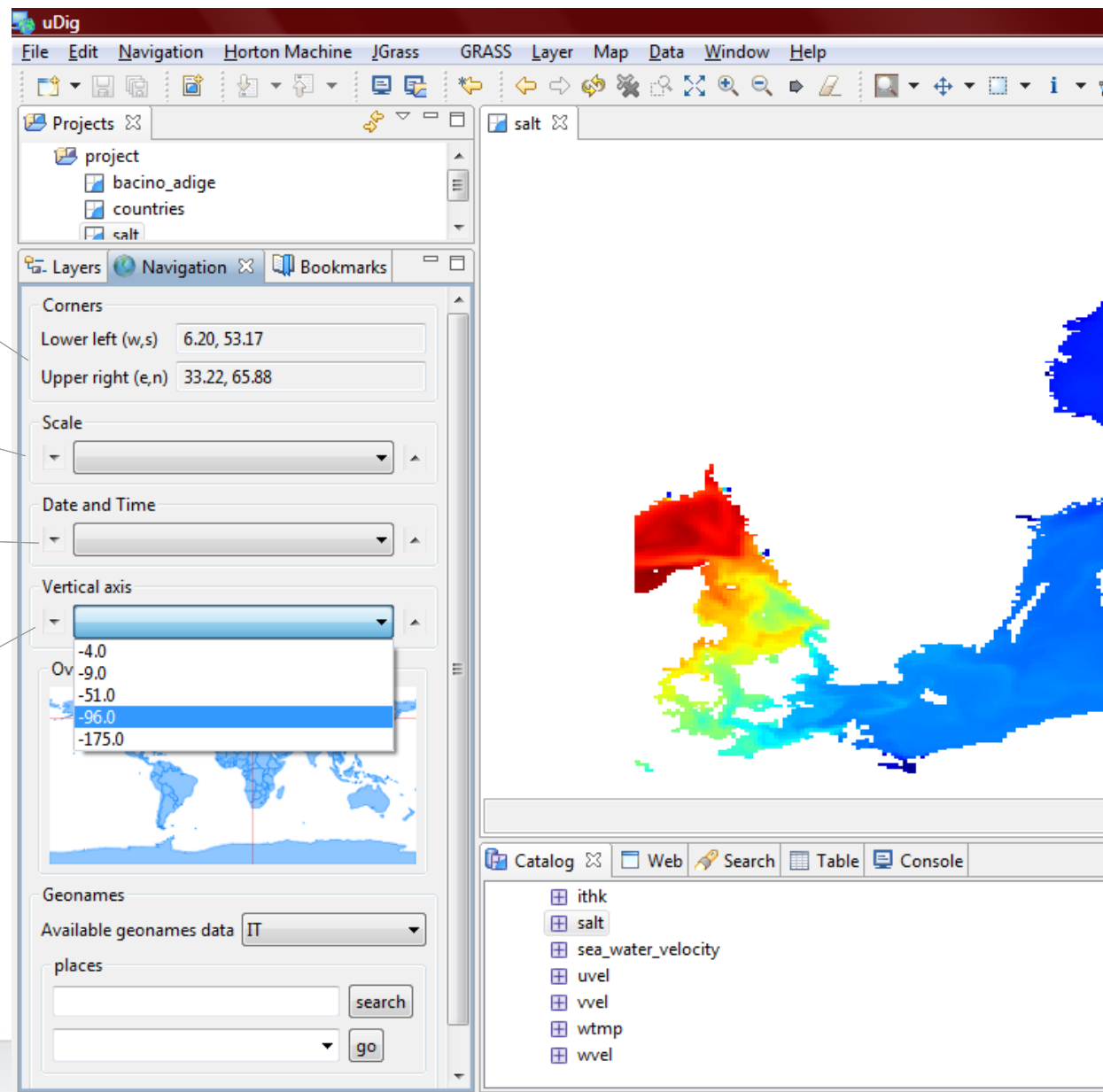
JGrass – the navigation view

current viewport boundaries
to copy/paste

quick setting of mapscales

setting of timestamp of
the layer if supported

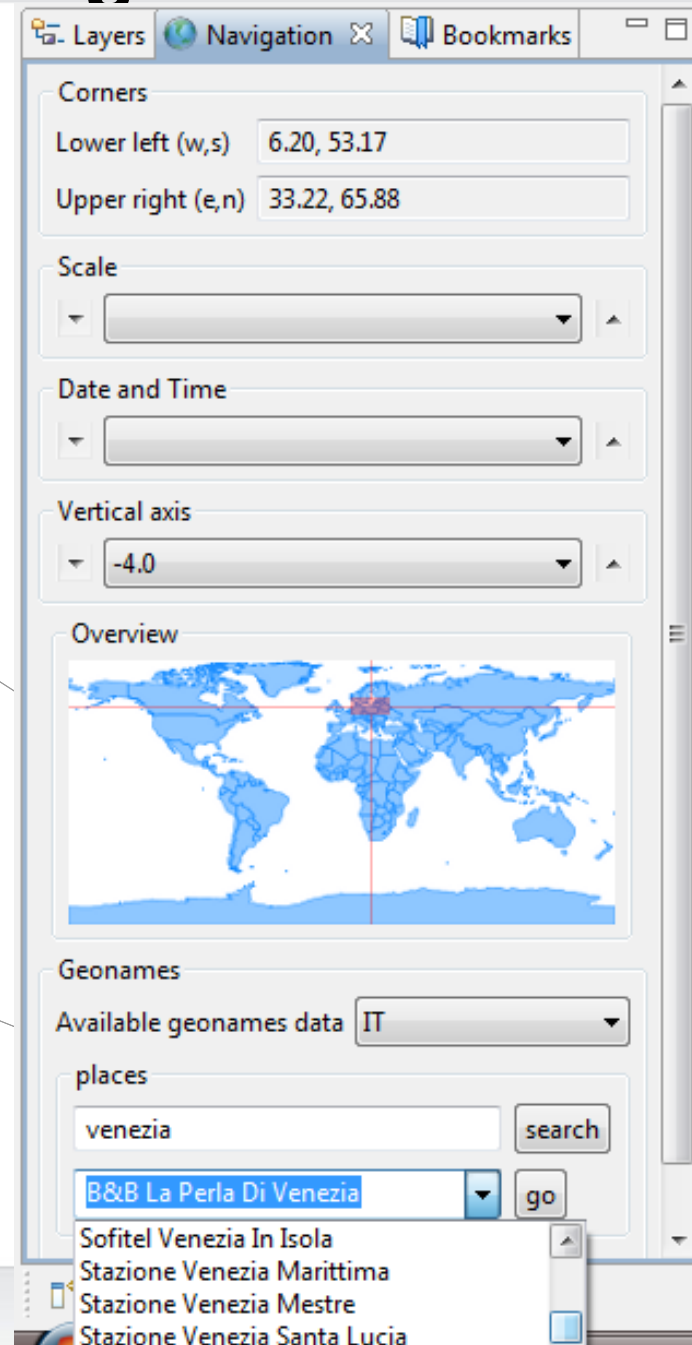
setting of vertical coordinate
of the layer if supported



JGrass – the navigation view

world view showing the bounds of the current visualized viewport

support for browsing of/zooming to geonames (<http://www.geonames.org>)



JGrass – centralized repository with Ramadda

RAMADDA (Repository for **A**rchiving, **M**anaging and **A**ccessing **D**iverse **D**Ata) is a development effort of the Unidata Program Center. RAMADDA is freely available and provides a **publishing platform, content management system and collaboration services for Earth Science data.**

Even if Ramadda is a quite new project, it has been proven to be a good choice for centralization of results of simulations directly from within JGrass.

Ramadda can be found at:

<http://www.unidata.ucar.edu/software/ramadda/index.html>

JGrass – centralized repository with Ramadda

Ramadda as seen as a j2ee web application

The screenshot shows a web browser window with the title "HydroloGIS Morpheo Data Repository - Group". The browser's address bar shows "HydroloGIS Morpheo Data Reposi...". The page header includes the "HydroloGIS Environmental Engineering" logo and the title "HydroloGIS Morpheo Data Repository - Group". A menu bar contains "File | Edit | View".

Links:

- Main Repository
- Search
- Admin

- Data Cart
- Logout
- Andrea Antonello
- Help

Favorites:

- Main Repository

Main Repository

Information

Created by: [Andrea Antonello](#) @ 2009-09-08 13:55:55 GMT

Date: 2009-09-08 13:55:55 GMT

Type: Group

Groups

- documents
 - client code
 - call_4_papers.odt
 - rasterlite-how-to
 - hydrologis logo
- morpheo
 - cam_i_0000.nc
 - water.nc
 - water_surf.nc

JGrass – centralized repository with Ramadda

For netcdf files great support for metadata browsing and editing

The screenshot displays the HydroloGIS Morpheo Data Repository interface. The main content area shows the entry for a netcdf file named 'water_surf.nc'. The interface includes a navigation menu on the left with links for Main Repository, Search, Admin, Data Cart, Logout, Andrea Antonello, and Help. Below the navigation menu is a 'Favorites' section with a link to the Main Repository. The main content area has a menu bar with 'File | Edit | View' and a breadcrumb trail 'Main Repository > morpheo'. The file 'water_surf.nc' is selected, and its description is 'test file created with JGrass'. The 'Information' section is expanded, showing a list of metadata properties under the 'Metadata' tab. The properties include contact information, geographic coordinates, conventions, distribution statement, references, and a comment.

HydroloGIS Morpheo Data Repository - Entry

File | Edit | View

Main Repository > morpheo

water_surf.nc

Description

test file created with JGrass

Information

Basic | **Metadata** | Variables

Property: Contact=andrea.antonello@gmail.com

Property: longitude_min=10.575944390272541

Property: longitude_max=10.738586384939904

Property: latitude_max=46.80318455068711

Property: latitude_min=46.64114081233529

Property: Conventions=CF-1.4

Property: Distribution_statement=IN NO EVENT SHALL MY COMPANY OR I

Property: References=http://www.hydrologis.com

Property: Comment=any comment here

JGrass – centralized repository with Ramadda

...support for variables browsing...

The screenshot displays the 'HydroloGIS Morpheo Data Repository - Entry' page. The browser tab is labeled 'HydroloGIS Morpheo Data Reposi...'. The page title is 'HydroloGIS Morpheo Data Repository - Entry'. The HydroloGIS logo and 'Environmental Engineering' tagline are visible in the top left. A navigation menu on the left includes 'Links:' (Main Repository, Search, Admin) and 'Favorites:' (Main Repository). The main content area has a menu bar with 'File | Edit | View'. Below it, the breadcrumb 'Main Repository > morpheo' is shown, followed by the file name 'water_surf.nc'. The 'Description' section contains the text 'test file created with JGrass'. The 'Information' section has three tabs: 'Basic', 'Metadata', and 'Variables', with 'Variables' being the active tab. Under the 'Variables' tab, two variable entries are listed: 'Variable: water_surface (m) water_surface' and 'Variable: single (-) single', with the second entry highlighted in light blue.

JGrass – centralized repository with Ramadda

...support for remote connection via opendap (JGrass can access that), for extraction of subsets of data and more.

The screenshot displays the HydroloGIS Morpheo Data Repository web interface. The browser tab is titled "HydroloGIS Morpheo Data Reposi...". The page header includes the HydroloGIS logo and the text "HydroloGIS Morpheo Data Repository - Entry".

On the left side, there are two sections: "Links:" and "Favorites:". The "Links:" section contains: Main Repository, Search, Admin, Data Cart, Logout, Andrea Antonello, and Help. The "Favorites:" section contains: Main Repository.

The main content area shows a file entry for "water_surf.nc". A context menu is open over this entry, listing various actions: Entry, Map, Grid Subset, CDL, Chat, Catalog, Dif-XML, Dif-Text, OpenDAP, RSS Feed, Graph, and View in IDV. The "View" menu item is currently selected.

Below the context menu, the following metadata is displayed:

- Created by:** [Andrea Antonello](#) @ 2009-09-08 14:19:27 GMT
- Resource:** [water_surf.nc](#) 3337000 bytes
- Date Range:** 2008-12-31 23:00:00 GMT ⇨ 2009-01-01 00:15:00 GMT
- Type:** File
- Data Type:** netcdf

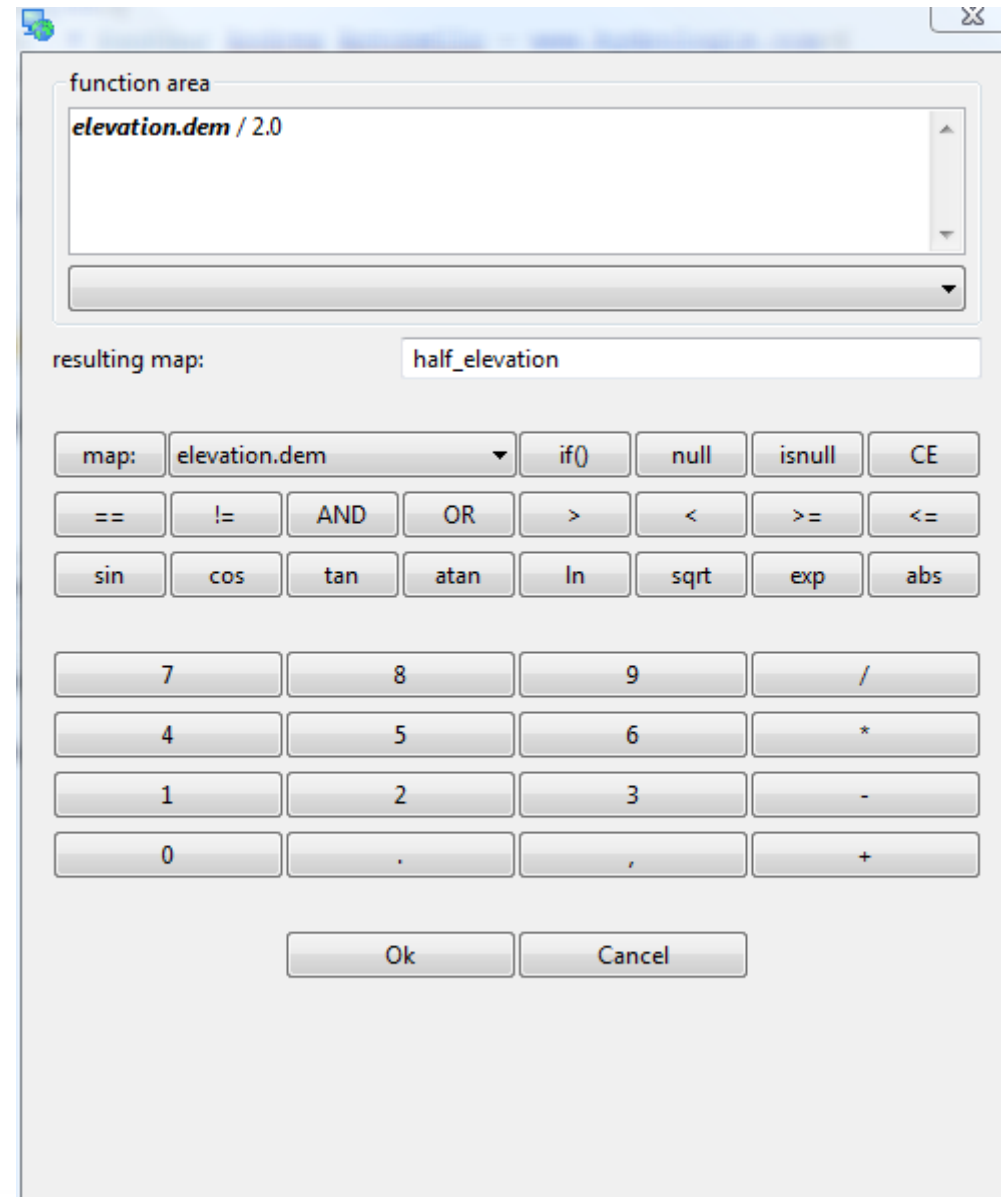
JGrass – raster mapcalculations

r.mapcalc: from GRASS over to Jiffle

Why a substitute to the GRASS

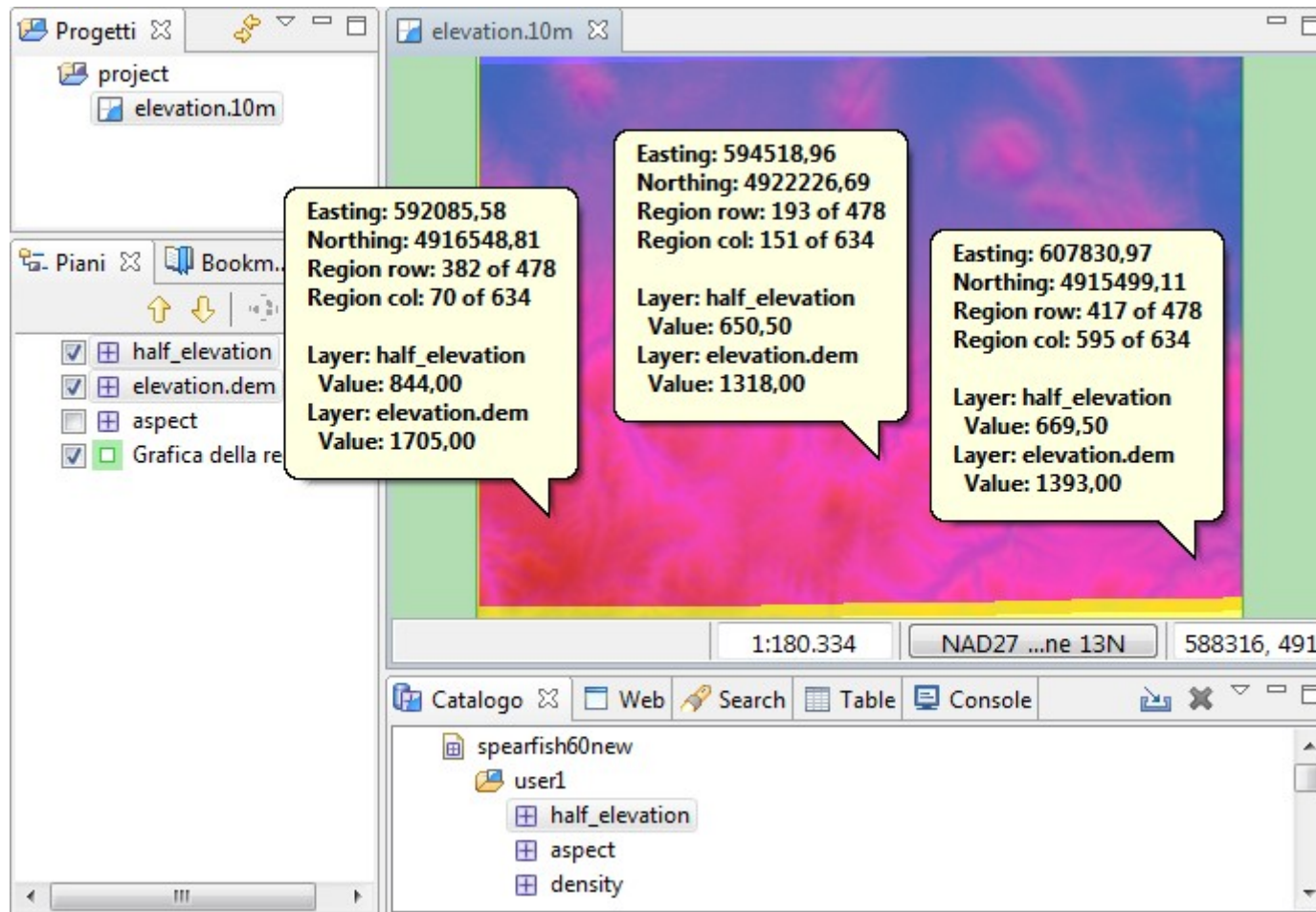
r.mapcalc?

- no control on r.mapcalc, since executed in runtime
- gave us huge random failures
- **we want support for tiling**
- jiffle is a pure java implementation based on jai and imageio



JGrass – raster mapcalculations

jiffle mapcalculations: an example



JGrass – raster tiling

With the average of lidar datasets at high resolutions, it was getting difficult to do certain analyses on large datasets.

So we expanded RAM, expanded RAM and expanded RAM...

Then the JGrass raster driver to imageio in order to be able to read the data tiled.

JGrass – new models

A large project was done with JGrass to handle the water household on the second river in Italy (Adige) taking into consideration special points like dams, intakes, offtakes, artificial channels. The responsible authority agreed to leave the created code under open source inside JGrass.

This added a bunch of new models to the JGrass modeling library as for example:

- meteorological interpolation models (h.jami, h.kriging).
- energy balance models (h.eicalculator, h.energybalance).
- discharge models (h.adige, h.santgeo).

ALL THESE MODELS IMPLEMENT THE OPENMI STANDARD

JGrass – new models

Example of eclipse forms in the h.energybalance model: inputs

basins_passirio_width1 h.energybalance energy_balance_2005.jgrass

5 warnings detected

▼ Pannello principale
 In questa sezione sono definiti alcuni parametri del modello.

Data inizio [yyyy-MM-dd]
 Data fine [yyyy-MM-dd]
 Passo temporale [mm]
 Percorso file output ...

AVO: Non è stato fornito un valore per AVO, si usa il valore di default di 0.85
 Percorso al safe point da leggere: Non sono state specificate condizioni iniziali da leggere
 Mostra il log per bilancio di massa: I dati di log non verranno visualizzati nella Console di JGrass
 Densità sui ghiacciai: Non è stato fornito un valore densità della neve sui ghiacciai, verrà usato il valore di default di 800 kg/m3
 AIRO: Non è stato fornito un valore per AIRO, si usa il valore di default di 0.65

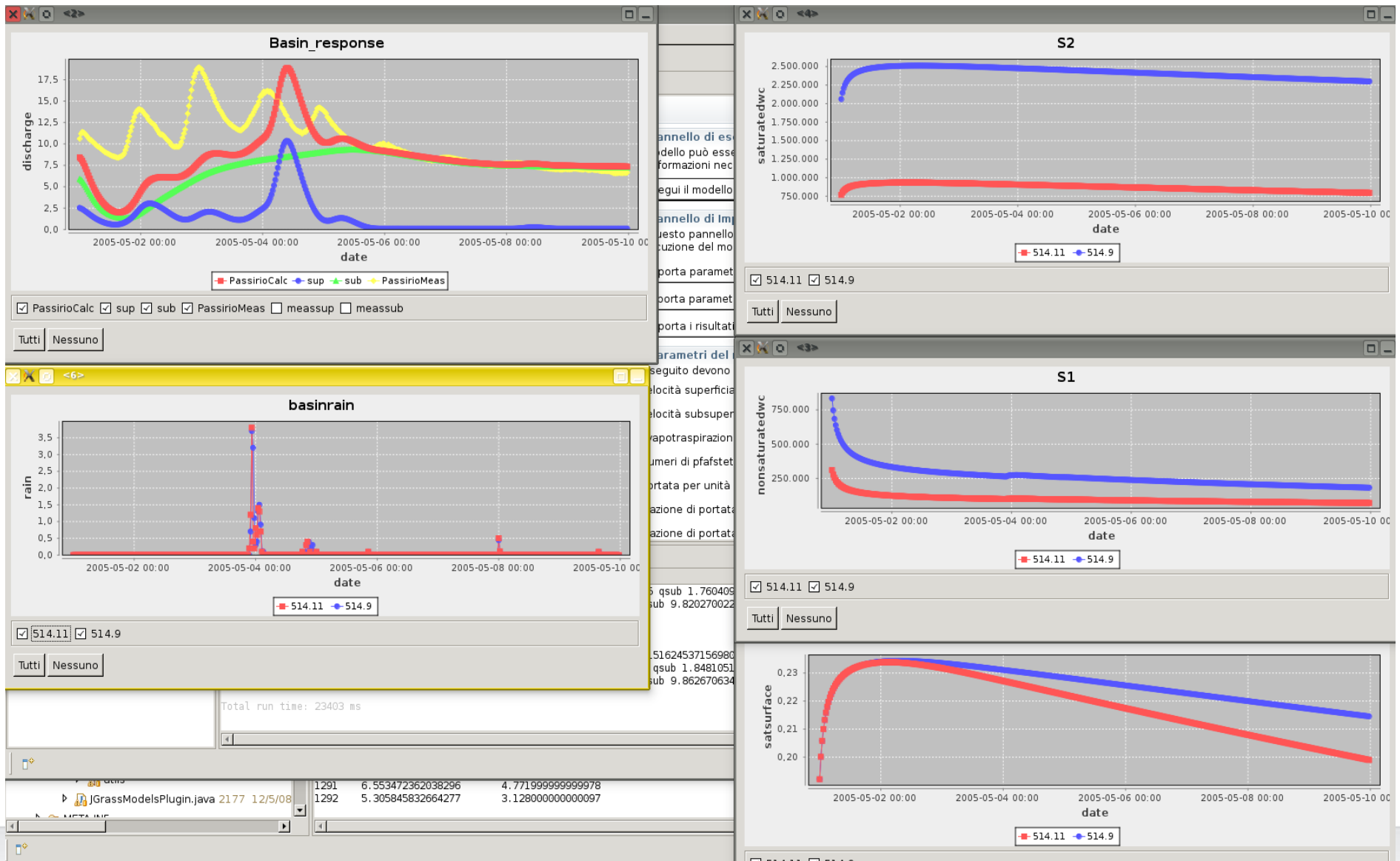
▼ Sezione dei parametri
 In questa sezione sono definiti alcuni parametri del modello.

Campo degli id dove interpolare	<input type="text" value="netnum"/>
Campo con uso suolo	<input type="text" value="uso_reclas"/>
SWE iniziale	<input type="text" value="0"/>
Valore per i ghiacciai	<input type="text" value="15"/>
AVO	<input type="text" value=""/>
AIRO	<input type="text" value=""/>
Densità sui ghiacciai	<input type="text" value=""/>
SWE ghiacciai	<input type="text" value="2000"/>
Sr	<input type="text" value="0.007"/>

▼ Sezione dei safe point

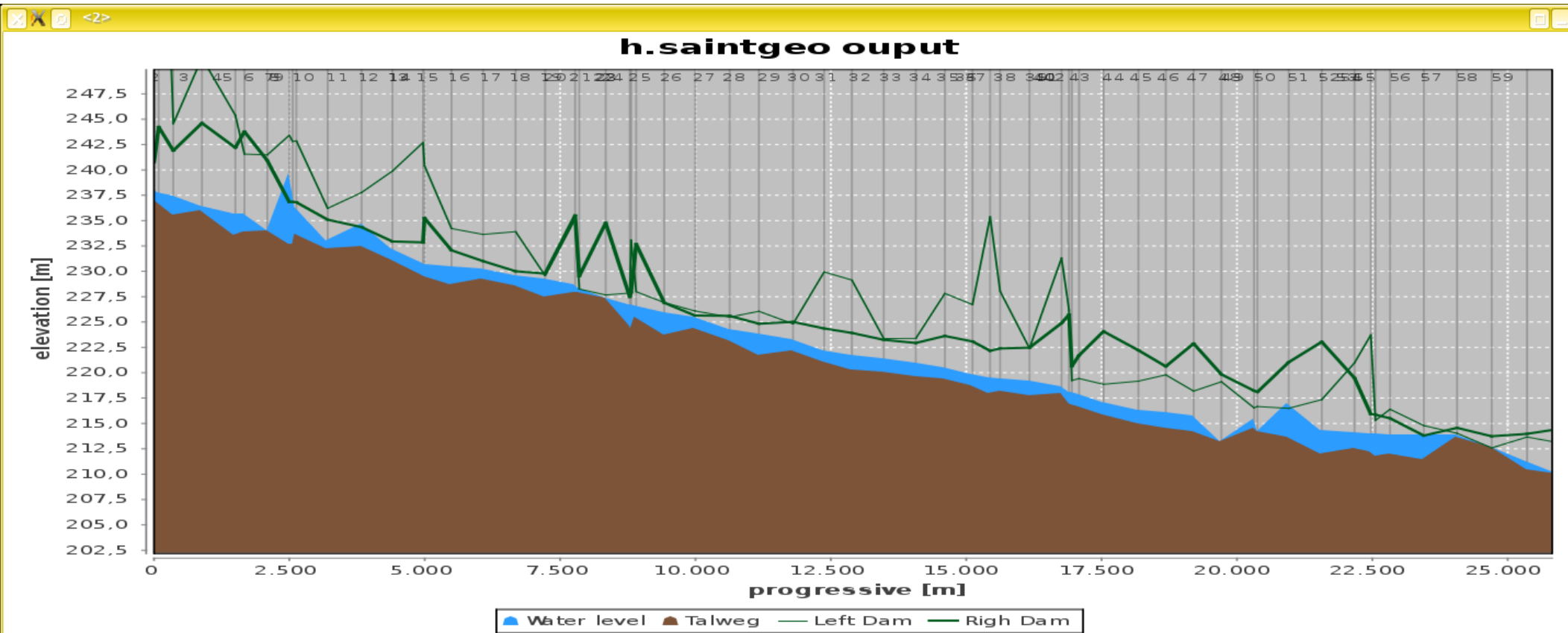
JGrass – new models

Example of the hydrologic model h.adige: results in progress



JGrass – new models

Example of the hydraulic model h.saintgeo: results in progress



Where are we heading to?

JGrass – a new strategy: towards OMS

The **Object Modeling System OMS** is a modular modeling framework that uses an open source software approach to enable all members of the scientific community to address collaboratively the many complex issues associated with the design, development, and application of distributed hydrological and environmental models.

OMS is pushed by the USDA (American Department of Agriculture).

OMS can be found at: <http://www.javaforge.com/project/omslib>

JGrass and OMS: what will happen to OpenMI?

A big effort has been done in the last years to bring all the models contained to OpenMI compliancy. There are several main issues that pushed the decision to migrate towards OMS:

- OpenMI forces modelers to use a quite restrictive API
- OpenMI is currently proposing its version 2, which from 1.4 introduces several changes. Migrate to that would require an enormous effort

JGrass and OMS: what will happen to OpenMI?

On the other hand:

- OMS already contains a set of components that are free and open sourced, and also already well tested at the USDA, which would come as a present to JGrass. OpenMI still doesn't have any open source components and seems to be focused on few proprietary applications
- OMS is an annotation based modern modeling framework that really focuses on adding few overhead to the modeler
- the OMS team is working on a wrapper to generate OpenMI code from OMS models

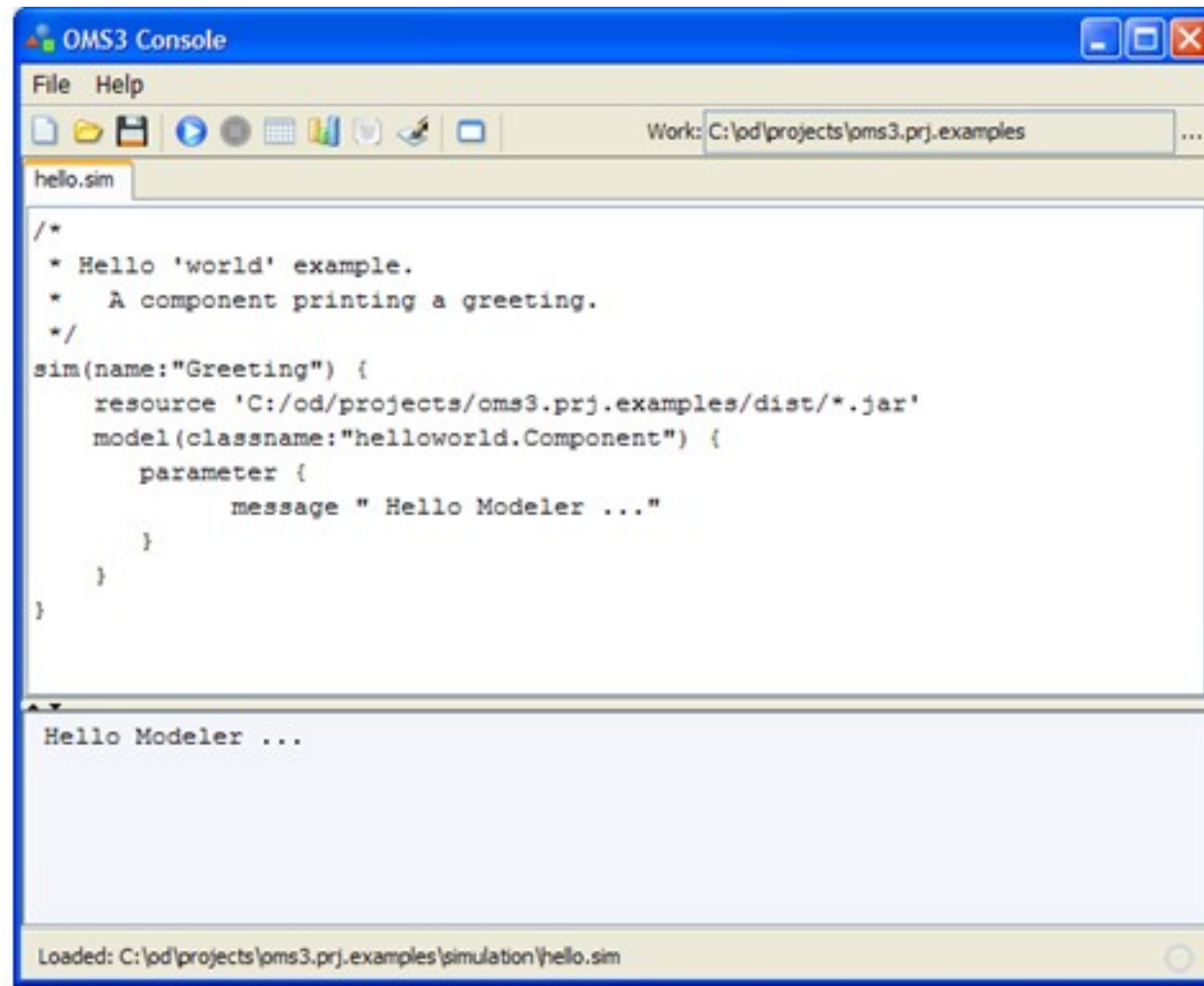
OMS: an annotations based framework

OMS minimizes the burden on a component/model developer to build code into the framework **by not imposing an API**. (I know everyone claims it, but believe me, this time it is true)

```
package helloworld;
import oms3.annotations.*;

public class Component {
    @Role(Role.PARAMETER)
    @In
    public String message;

    @Execute
    public void run() {
        System.out.println(message);
    }
}
```

The screenshot shows a window titled "OMS3 Console" with a menu bar (File, Help) and a toolbar. The work directory is "C:\od\projects\oms3.prj.examples". A file named "hello.sim" is open, displaying the following XML-like code:

```
/*
 * Hello 'world' example.
 * A component printing a greeting.
 */
sim(name:"Greeting") {
  resource 'C:/od/projects/oms3.prj.examples/dist/*.jar'
  model(classname:"helloworld.Component") {
    parameter {
      message " Hello Modeler ..."
    }
  }
}
```

The console output shows "Hello Modeler ...". The status bar at the bottom indicates "Loaded: C:\od\projects\oms3.prj.examples\simulation\hello.sim".

OMS: other advantages

With OMS a bunch of important features come into JGrass's modeling system:

- Components always execute **multi-threaded**. If the data flow allows it, the models are executed in parallel.
- **Runtime flexibility** for simulation execution. Models can be executed in different environments that scale from a notebook to a computing cluster or even a cloud such as Amazon's Elastic Computing Cloud (EC2).

This is extremely important to JGrass, since we often need to execute models that run for hours and days, and want to exploit high performance computers or clusters.

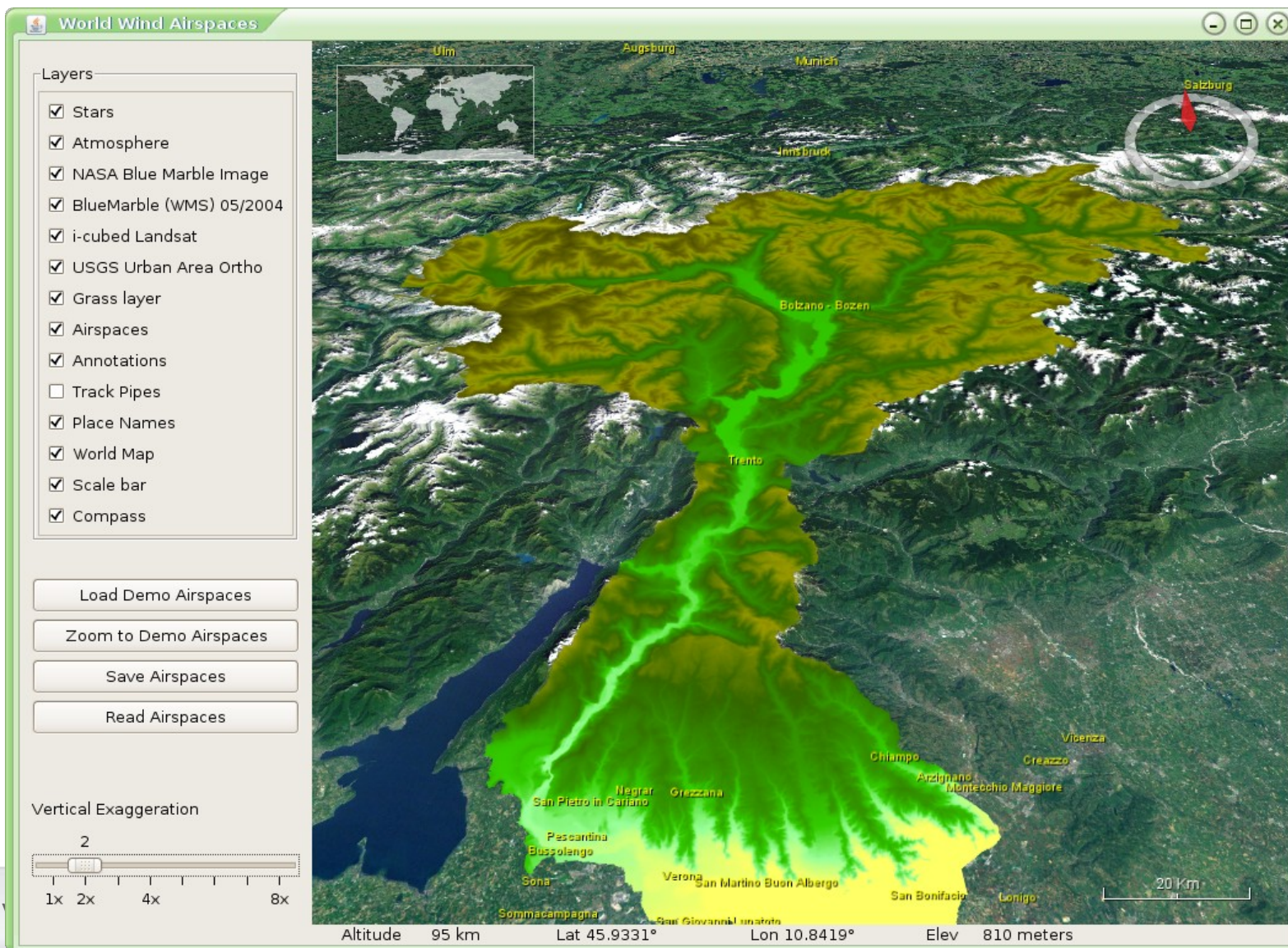
OMS: other advantages

With OMS a bunch of important features come into JGrass's modeling system:

- Components always execute **multi-threaded**. If the data flow allows it, the models are executed in parallel.
- **Runtime flexibility** for simulation execution. Models can be executed in different environments that scale from a notebook to a computing cluster or even a cloud such as Amazon's Elastic Computing Cloud (EC2).
- **Integration with JNA** (same as JGrass) for native code access. Java Native Access (JNA) integration that now supports all versions of FORTRAN, C, and C++ on all major architectures in 32 and 64 bit. FORTRAN and C/C++ programmers can continue to use their respective tools to create components
- The OMS modeler environment **bases on Groovy** scripting language, exactly as JGrass's console does
- Loading of models **libraries at runtime**

Prototyping JGrass I/O into Nasa World Wind

A small test has been done to link the JGrass I/O drivers into the NWW tiling mechanics to use it to tile and cache GRASS rasters the NWW way.



Prototyping JGrass I/O into Nasa World Wind

A small test has been done to link the JGrass I/O drivers into the NWW tiling mechanics to use it to tile and cache GRASS rasters the NWW way.





JGrass is a Free Software system which has been developed by HydroloGIS and CUDAM since the year 2003. The original community however is seeking for creating around JGrass an **ecosystem of co-developers and users**. In fact from the beginning JGrass was designed to serve the community, looking at a better interface for GRASS, and already made a further step in this direction **joining the uDig** community. **Beegis** is a new cooperation between HydroloGIS and the University of Urbino, that can serve as an example for other Institutions and people.