wxPython-based GUI for GRASS GIS

Martin Landa¹, Cristina Moretto¹,
landa.martin@gmail.com, moretto@fbk.eu

Michele Zanolli², Luca Manganelli²,
michele_zanolli@comune.trento.it, luca_manganelli@comune.trento.it

Markus Neteler³
neteler@cealp.it

¹ Fondazione Bruno Kessler,
http://www.fbk.eu
² Comune di Trento,
http://www.comune.trento.it
³ Fondazione Edmund Mach – Centro di Ecologia Alpina,
http://www.fmach.eu

Abstract. The plan for a native GRASS Graphical User Interface (GUI) finds its origin in the project GFOSS – TN, fruit of the collaboration between FBK (Fondazione Bruno Kessler) and the “Information System Service” of the TRENTO municipality. GFOSS – TN derives from the need to define and to monitor a migration process in which GIS instruments and functions are transferred in Open Source environment.

Native GUI is one of the key points for the future development of GRASS GIS. It is crucial for GRASS user and especially connected to newcomers. GRASS offers high capability to analyze and manipulate spatial data, and covers many application areas of GIS projects. However, it shows an incomplete and scarcely intuitive graphical user interface, which causes slow performance of many options such as layout of maps. The position of GRASS on the market does not correspond to its capabilities which the software can offer to the end-user.

One of the main obstacles to its adoption by GIS professionals, are the issues of ease of use and portability. A large number of GIS professionals use MS-Windows and Macintosh operating systems. Until recently, GRASS run on these platforms not without problems. Additionally, the user interface of GRASS has lagged behind modern GUI design, making it much harder to operate.

The standard GUI is written in Tcl programming language using Tk graphical toolkit. In the last years the limitations of Tcl/Tk toolkit appeared to be prohibitive for the future development. The new generation of GRASS GUI is written in the Python programming language using wxPython – a blending of the wxWidgets library for Python. wxPython GUI will be introduced in stable release GRASS 6.4.0 (the main release of 2008).

The GUI is composed of two main components: Layer Manager and Map Display Window. The Layer Manager allows users to run different GRASS modules from a menu, includes map layer management, integrated command-line prompt, and command output window. Map Display Window integrates basic tools for zooming, panning, data querying, decorations (north arrows, barscale, etc.). Additional tools like Digitization or Georectification tool are also available.
The future development will be focused on the integration of OpenGL. As result, Map Display Window will support 3D rendering including 3D raster (voxels) and 3D vector data.

A Map Composer, a tool for hardcopy map outputs is planned to be developed in the first half of 2008. Currently, the tools for creating hardcopy maps (map layout) are limited in GRASS because its focus on modeling and spatial analysis. GRASS gives the user ability to add only a very simple and standardized legend, north arrow and scale to display to the graphics monitor and then export the display to an external image file such as a png. These map features are very basic and cannot be customized. The goal is to allow users to prepare simple cartographic outputs comparable e.g. with ESRI’s ArcGIS map layout functionality.

Abstract. La realizzazione di una Interfaccia nativa per GRASS nasce nell’ambito del progetto GFOSS-TN, frutto di una collaborazione tra la Fondazione Bruno Kessler (Povo-TN) e il Servizio Sistema Informativo del Comune di Trento. Il progetto nasce dall’esigenza di definire e monitorare un processo di migrazione in cui strumenti e funzionalità GIS sono progressivamente trasferiti in ambiente Open Source.

Uno dei principali ostacoli all’adozione di GRASS GIS, da parte di professionisti GIS, si riscontra proprio nella ridotta usabilità e nei problemi di portabilità del software. Un largo numero di utenti utilizzano infatti sistemi operativi come MS Windows e Macintosh per i quali GRASS, pur presentando notevoli funzionalità nell’analisi e manipolazione dei dati, offre una interfaccia incompleta, scarsamente intuitiva e poco performante in molte opzioni come il layout map.

L’attuale GUI è scritta usando il toolkit grafico Tcl/Tk le cui limitazioni si sono rivelate compromettenti per lo sviluppo futuro mentre la nuova interfaccia (introdotta nella stable release di GRASS 6.4) è scritta in Python usando wxPython, una libreria grafica OS che permette la realizzazione di GUI multipiattaforma.

Due saranno le componenti principali: il Layer Manager e la Map Display Window. Il Layer Manager permetterà agli utenti di eseguire da menu diversi moduli come la gestione dei tematismi e una linea di comando integrata. La Map Display Window integrerà strumenti di base come zoom, pan e query dei dati. Saranno inoltre resi disponibili tool addizionali per la digitalizzazione e la georettifica.

Gli sviluppi futuri si focalizzeranno sull’integrazione di OpenGL che permetterà di supportare rendering 3D sia per dati raster (voxel) che vettoriali. È in programma anche lo sviluppo di Map composer, un tool per la stampa di qualità che permetta di ottenere risultati comparabili con le funzionalità di map layout di ESRI/ArcGIS.

Keywords: GIS, GRASS, GUI, development, wxPython

1 Introduction

Trento is a medium-sized municipality of 112,000 inhabitants, the capital of the Province of Trento, one of two provinces which make up Italy’s region of Trentino-Alto Adige/Südtirol, which is an autonomous region as well. The municipality offers different types of services to their citizens with respect to public administration. One of them is “Servizio Sistema Informativo” (information service
The main aims of the “Sistemi, informatica distribuita e territorio” office are:

1. Manage and maintain the Trento cartographic data base,
2. Offer to the other services the instruments for data consulting and revision,
3. Reveal to the citizen part of information using web applications.

In this context, the most important tasks is the administration of the “Piano Regolatore Generale” (City Plan), the toponomy, the cadastre and the registrar’s office.

The core team of the SIT is made up of six technicians; the main software system adopted to use and to elaborate cartographic data is based on proprietary software.

Till 2005 also the webGIS service was based on a proprietary software and its administration was delegated to an external venture. In 2005, the region published a regional resolution\(^1\) to suggest guidelines for interoperability and to promote the use of Free Software. As result, the Trento Municipality started to pay attention to Open Source software, and to start thinking to definitely leave proprietary software in the future.

The first step was to move the administration of the webGIS inside of the SIT. It was giving the advantage of maintenance cost reduction of the service \(^7\), and the possibility to speed up the procedures of data updating and enrichment. The new portal is not based on a proprietary solution but on the Open Source UMN MapServer.

The second step was dedicated to migrate the desktop GIS applications.


**Fig. 1.** Software operating costs (customer).
At that point, Trento Municipality decided to collaborate with FBK (Fondazione Bruno Kessler), a research institute in Trento. For a long time, FBK supported already the GRASS GIS development, the world’s leading Open Source GIS.

The contribution of the SIT in the development consists of:

1. Financing the GFOSS-TN project dedicated to the development of the GUI for GRASS GIS (and the GRASS development in general),
2. Operate as power users and beta tester of the new implementation.

The agreement has been made official through the publication of two resolutions from the municipality government, with the first in 2005 (GFOSS-TN1) and the second in 2007 (GFOSS-TN2). These two resolutions engaged FBK for five years and at a total of 43,000 Euro dedicated to the development in order to meet the SIT’s demands.

The Municipality invests annually 94,400 Euro for proprietary licenses, with additionally 15,900 Euro for maintenance.

According to the resolution, the municipality decided to migrate to Free Software for the following reasons:

1. Reasonable reduction costs of software licenses and purchase of maintenance,
2. Use of open standards to facilitate the data exchange with other agencies and inner organizations,
3. Possibility to develop a webGIS while avoiding dependencies on external ventures,
4. Possibility to promote within the Municipality a software culture not correlated to a particular software house,
5. Possibility to build a standardized system of data distribution and spread with other local Public Administration of the Trento Province,
6. Diffusion of Open Source culture within the Public Administration.

2 WxGUI in GRASS GIS

The original native GUI (Graphical User Interface) for GRASS [1] was written in Tcl programming language using the Tk graphical toolkit. The first prototype was called “TCLTKGRASS” (1994/1999), from that evolved the “GIS Manager” (2004), additionally a new v.digit and “NVIZ” user interface are written in Tcl/Tk as available in the last version GRASS 6.3.0. Limitations of the Tcl/Tk toolkit made further GUI development difficult and less attractive for potential contributors. In 2006, it was decided to leave Tcl/Tk behind and to design new native GUI for GRASS from scratch using a different graphical toolkit.

The main project goals of the new GUI are:

- Portability, enabling GRASS to be fully functional on the GNU/Linux, Unix, MacOSX and MS-Windows operating systems (fig. 2);
- Ease of Use, providing all the features commonly found in state of the art GIS applications;
- Extensibility, all tools integrated into GUI including digitisation, georectification, image classification, etc.
As result, the improved native GUI should help to improve the diffusion of GRASS use in productional environments as opposed to research environments. The new styled GUI with improved scripting facility (focused on Python programming language) should reduce the time of project implementation and also significantly simplify GRASS usage for newcomers.

For development, wxWidgets library is used (formerly known as wxWindows) [2]. WxWidgets is well-documented, well-known and a widely used graphical toolkit. The crucial characteristics connected to the GRASS GUI development are:

- Powerful Python binding (know as wxPython);
- Use of native platform SDK (native look and feel); this means that a program compiled on MS-Windows has the look and feel of any MS-Windows program; likewise, when compiled on a GNU/Linux machine, it shows the look and feel of a GNU/Linux program so that users feel at home.
- Free OpenGL widget (in contrast to Qt which lacks this functionality) which is required for future developments (e.g., NVIZ replacement).

For the real development Python cross-platform wrapper for the wxWidgets has been chosen which is known as wxPython [3]. The decision to use Python as the operating platform for this effort came rather naturally when thinking about modularity, portability, extensibility and reliability. Python as “easy-to-learn”, object oriented, currently “very popular” language should enable more people to actively contribute to the development in contrast to Qt which requires C++ (or Tcl/Tk used for the old GUI). Moreover, wxPython is currently actively developed and maintained.

The GUI development based on the wxPython graphical toolkit started in the end of 2006. The wxPython-based GUI (wxGUI) is included as an experimental prototype in the last release GRASS 6.3.0 (first technology preview version for the planned main stable release of GRASS 6.4.0 in 2008). wxGUI is planned to be default GUI for the new generation of GRASS – GRASS 7.
Detailed information are available on the GRASS-Wiki [5].

The core components of wxGUI are:

- **Layer Manager** (fig. 3) which allows users to run different GRASS modules from menu, includes map layer management, integrated command-line prompt, and command output window.
- **Map Display Window(s)** (fig. 4) which integrates basic tools for zooming, panning, data querying, decorations (north arrows, barscale, etc.). This component replaces the old X11-based GRASS monitors. The user is allowed to start various Map Display Windows during one session. The Layer Manager registers open Map Display Windows using different tabs.

![Fig. 3. Layer Manager (map layer management and command output area).](image)

There are already several additional GUI components implemented:

- Attribute Table Manager (fig. 5),
- Vector digitization tool (fig. 8),
- Georectification tool (fig. 6),
- Location wizard,
- Profile analysis tool (fig. 7),
- Histogram tool for raster maps and images.

One of the key GUI components, the “digitization tool” (fig. 8) is fully integrated into GUI and replaces the current Tcl/Tk-based v.digit module. The functionality of the former v.digit module has already been re-implemented.

Moreover, new features (requested by the GIS department of Comune di Trento) have been implemented:

- Display vertices, animate movement of vector object by mouse,
- Select vector objects by box,
- Select vector objects by query (line length, dangles),
- Snap to node or vertex,
Fig. 4. Map Display Window.

Fig. 5. WxGUI (Layer Manager, Map Display Window and Attribute Table Manager).

- Snap to the vector objects (node or vertex) from background vector map layer(s),
- Unsplit (remove pseudo-nodes) – merge selected vector lines,
- Connect selected lines (undershooted lines),
- Break selected lines,
- Copy vector objects from background vector map layer(s),
- Z bulk-labeling (automated z-coordinate assignment to selected vector lines, e.g. vector contours),
- Check for duplicates when selecting vector features by bounding box (fig. 9),
- Undo function of digitization.
2.1 Future developments

The last key GUI component currently missing in wxGUI is an integrated 2.5/3D visualization tool (i.e., NVIZ replacement). The Map Display Window component currently supports only displaying data in 2D, the development is focused on the integration of OpenGL [4]. OpenGL is a vendor neutral, window system independent, 3D rendering application programming interface (API) that provides on-screen and offscreen access to the graphics hardware. As result, the Map Display Window will support
3D rendering including 3D raster (voxels) and vector data. Functionality will be based on the current NVIZ module.

Many other components or sub-components have to implemented or improved including *interactive command-line prompt*, module search engine (to allow users to find a GRASS module based on the given keywords), *on-the-fly projection* automatically convert and overlay spatial data in different projections. Another issue is to rewrite and improve X11-monitor dependend `i.class` and related imagery modules and integrated them into wxGUI.
The tools for creating hardcopy maps (map layout) are yet quite limited in GRASS because of its focus on modeling and spatial analysis. GRASS gives the user ability to add only a very simple and standardized legend, north arrow and scale (modules like d.barscale,d.legend) to display these elements on the graphics monitor and then export the display to an external image file such as a in PNG format. These map styling features are yet very basic and cannot be customized.

Beside that, GRASS contains a specialized module for hardcopy map outputs (ps.map) which works only non-interactively. Preparation of a control file with mapping instruction for ps.map can be for non-experienced user a very time consuming task. The module lacks a graphical front-end which would enable the user to compose a map layout interactively. A GUI front-end for ps.map is currently under development.

The next step towards better cartographic tools in GRASS requires improving or basically rewriting related cartographic GRASS modules such d.barscale, d.legend, d.vect, etc. At the end, the Map Display Window component will be used also for map layout preparation. Then content of map canvas will be simply exported to Postscript format using GRASS PS driver (or other formats such PNG, PDF or SVG). First of all, the general display architecture needs to be improved to support floating-point coordinates. An integration with proposed OSGeo Cartographic Library [6] is suggested.

3 Conclusion

Switching from Tcl/Tk to wxPython graphical toolkit was a crucial step for future GRASS GUI development. WxPython is the most actively maintained wrapper for the wxWidgets library.

The GRASS wxPython-based GUI is currently being developed by a group of programmers. It appears that Python as an “easy-to-learn” programming language enables more people to actively contribute to the development process (in comparison with the Tcl programming language where development did not scale well).

The improvements in the graphical user interface will help to bring more GIS users and professionals into the GRASS community, in turn leading to further advances in GRASS and GIS in general and ultimately a benefit for the public.

Reference

   http://grass.osgeo.org/wiki/WxPython-based_GUI_for_GRASS.