Workshop
on Open Source Desktop GIS

IIITMK, Technopoark
Trivandrum, Kerala, India

Prof. Dr.-Ing. Dietrich Schröder
University of Applied Sciences Stuttgart, Germany
dietrich.schroeder@hft-stuttgart.de

October 2008

Table of contents

Part A: Introduction

Part B: Case Study

Part A: Introduction

1. Open Source Desktop GIS

A **geographic information system (GIS)** is an information system for capturing, storing, analyzing, managing and presenting data which is spatially referenced (linked to location).

GIS applications are tools that allow users to create interactive queries (user created searches), analyze spatial information, edit data, maps, and present the results of all these operations. Geographic information system technology can be used for scientific investigations, resource management, asset management, archaeology, environmental impact assessment, urban planning, cartography, criminology, marketing, and logistics to name a few.

*A **desktop GIS** is a mapping software that is installed onto and runs on a personal computer and allows users to display, query, update, and analyze data about geographic locations and the information linked to those locations*.[1]
That is, the GIS software is not executed on a server and remotely accessed or controlled from or by a different computer. Another definition focus on the functionality:
A desktop GIS is a GIS program with an interactive GUI and reduced GIS functionality, mainly used for visualizing of GIS data or developed only for special applications.[2]
Of course the borders of the different types of GIS is blurred, there are data viewers which offer additional functionality like Thuban[1] or mezoGIS[2], where as nearly all desktop GIS nowadays offers the possibility of consuming OGC services like WMS or WFS, so they are using resources from the Internet. On the other hand, workhorses like GRASS are still running on one single personal computer but offering full GIS functionality.

Over the last few years the world of free and open source geospatial software has experienced some major changes. For instance, the website FreeGIS.org currently lists more than 330 GIS related projects. With the broad use of non-proprietary and open data formats such as the Shape File format for vector data and the Geotiff format for raster data, as well as the adoption of OGC standards for networked servers, development of open source software continues to evolve, not only for web and web service oriented applications, but also desktop GIS have some kind of renaissance. Well-known open source desktop GIS software includes GRASS GIS, Quantum GIS, uDig, OpenJUMP, gvSIG and many others. Despite the diversity of projects an exchange among projects can be clearly noticed. This holds true especially for the use of software libraries that enable data format conversions (e.g. GDAL/OGR, GeoTools), provide coordinate projections (Proj4), or offer basic geometric algorithms (e.g. GEOS, JTS). These libraries are used by open source and commercial software alike to provide basic functionality.

There are several dozen Open Source based Desktop GIS applications available to choose from, most of which are optimized for specific application areas. The relevant developments are mostly based on Java or C++ and provide a high degree of flexibility due to standardized interfaces and the capability to be deployed within service oriented architectures.

---

[1] ESRI *GIS Dictionary* [online]. http://support.esri.com/index.cfm?fa= knowledgebase.gisDictionary.search&searchTerm= desktop%20GIS

[2] Buhmann/Wiesel [2007]. GIS Report. Bernhard Harzer Verlag, Karlsruhe, Germany (in German)

Nevertheless, desktop GIS are powerful tools and but still require a solid background and good training to unleash its full potential. In the following the most important developments with an active developer and user community will be introduced.

Open source[3] is a development methodology, which offers practical accessibility to a product's source (goods and knowledge). Some consider open source as one of various possible design approaches, while others consider it a critical strategic element of their operations. Before open source became widely adopted, developers and producers used a variety of phrases to describe the concept; the term open source gained popularity with the rise of the Internet, which provided access to diverse production models, communication paths, and interactive communities. The following Open Source Definition is used by the Open Source Initiative (OSI) to determine whether or not a software license can be considered open source[4]:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

[3] see http://en.wikipedia.org/wiki/Open_source and http://en.wikipedia.org/wiki/GNU_General_Public_License
[4] see http://opensource.org/docs/osd

An **open source license** is a copyright license for computer software that makes the source code available under terms that allow for modification and redistribution without having to pay the original author. Such licenses may have additional restrictions such as a requirement to preserve the name of the authors and the copyright statement within the code. Most OSGIS are using either GPL or LGPL as licensing policy.

The **GNU General Public License** (GNU GPL or simply GPL) is a widely used free software license. The GPL is the most popular and well-known example of the type of strong copyleft license that requires derived works to be available under the same copyleft. Under this philosophy, the GPL is said to grant the recipients of a computer program the rights of the free software definition and uses copyleft to ensure the freedoms are preserved, even when the work is changed or added to. This is in distinction to permissive free software licences, of which the BSD licences are the standard examples. The distribution rights granted by the GPL for modified versions of the work are not unconditional. When someone distributes a GPL'd work plus their own modifications, the requirements for distributing the whole work cannot be any greater than the requirements that are in the GPL. This requirement is known as copyleft.

The **GNU Lesser General Public License** (LGPL) is a modified, more permissive, version of the GPL, originally intended for some software libraries.

2. The Jump Family

The Java Unified Mapping Platform (JUMP) is a GUI-based application for viewing and processing spatial data. It includes many functions common to other popular GIS products for the analysis and manipulation of geospatial data. The JUMP also provides a highly extensible framework for the development and execution of custom spatial data processing applications.

In 2002, as a project for the British Columbia Ministry of Sustainable Resource Management, Vivid Solutions Inc. created a software program to do automated matching ("conflation") of roads and rivers from different digital maps. The software team wisely made the program flexible enough to be used not just for roads and rivers, but almost any kind of spatial data: provincial boundaries, power-station locations, satellite images, and so on. The program was named JUMP (JAVA Unified Mapping Platform).

Based on the JUMP-framework several different applications have been developed, like OpenJump, deeJump, and gvSIG. All these application have been developed on Java and offer the possibility to add more functionality using plug-ins

2.1. OpenJump, deeJump, Kosmo

After the initial creation and deployment of JUMP, regular development of the program by Vivid Solutions stopped. However, the company continued offering support to the user community that had grown around JUMP, and provided information to developers that had begun to improve JUMP in small ways, or who had customized it to fit their needs.

It soon became evident that both the users and developers would benefit from a "unified" JUMP platform. This central or core platform would eliminate the compatibility issues that plagued the JUMP user community, and would give developers a platform on which to focus and coordinate their efforts. A number of the lead members from each team working with JUMP formed the JPP Development Committee, whose purpose was to guide and oversee this new unified platform. A name was chosen for this open source GIS program to be based on JUMP, "OpenJUMP"

So most of the features of JUMP are available in OpenJump, and meanwhile a lot of additional features have been added.

Open Jump

- reads and writes ESRI Shapefile, GML files, DXF and PostGIS
- reads raster files like TIFF, JPG, PNG and ECW
- save view to georeferenced rasters like JPG and PNG
- full geometry and attribute editing
- OpenGIS SFS (simple feature specification) compliant
- Geometry algorithms based on Java Topology Suite (JTS)
- a lot of third party plugins exists supports standards like WMS, WFS and SLD
- easy extensible GIS programming environment for own GIS-applications

On the very well organized OpenJump homepage[3] additional information can be found, including documentation and links to plug-ins. The focus of the plug-ins is on data loading and viewing of data (e.g connecting to PostGIS Oracle database or ArcSDE, GPS data, print, reproject vectors, etc.), where as spatial analysis are up to now not really exist.

Personal comments: It is more a less a container for plug-ins. The system itself offers only some basic tools for analysis.

deeJump [7] is a project of the German company lat/lon. The goal of this project is to add OGC compliant web functionality. It has some functions of OpenJUMP and has some further functionalities with respect to web standards. Since autumn of 2007 deeJUMP is based on OpenJUMP and deeJUMP functionality is added via plugins. Main parts of the implementation have been taken from the deegree server project of lat/lon, so the named it deeJump.

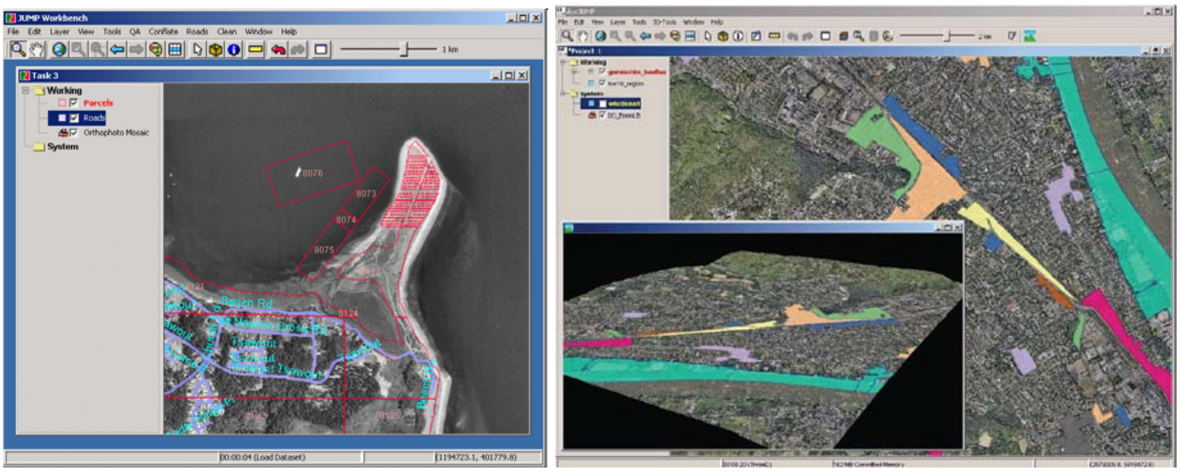


Figure 1: openJump / deeJUMP User interfaces

Personal comments: Plug-in for OGC services, no improvements for spatial analysis added.

The idea of the **Kosmo** [8] project is to develop a whole suite of GIS software including desktop, server, OGC client, and mobile. The ambitious project is still under development supported by the Spanish company SAIG, up to now most of the homepage and documentation only exist in Spanish.

2.2. gvSIG

gvSIG is a GIS that can handle both vector and raster data. It features basic editing tools for the creation and maintenance of vector or raster spatial data on a variety of file formats, including remote data sources. gvSIG is being actively developed in Java by IVER Tecnologías under the GNU General Public License (GPL). Its name is an abbreviation that stands for *Generalitat Valenciana, Sistema d'Informació Geogràfica*.

gvSIG was started in 2003 when the *Conselleria d'Infraestructures i Transports* (CIT) of Valencia in Spain proposed the development of software for the management of geographic information. The private enterprise IVER Tecnologías won the proposal and is now developing the software together with the Generalitat Valenciana and the Jaume I University of Castellón. So it is a Spanish driven project, which still can be noticed from the GUI, the documentation, and the homepage [4].

It is targeted at professional users of geographic information from the public sector, private industry sciences and education. It is developed as an Open Source project and currently undergoes the OSGeo Incubation process. The software provides internationalization modules and several languages are already supported.

The plug-ins for gvSIG include SEXTANTE [5]. SEXTANTE is an analysis suite of more than 160 tools that has been developed by the Regional Government of Extremadura. It has provided gvSIG with both raster and vector geographical analysis capabilities, geo-statistics, vegetation indexes, profiles and hydrological analysis, fuzzy logic, point pattern analysis to name but a few of the implemented functions. All the extensions are based on a set of foundational classes, specially conceived to ease the implementation of algorithms for geographical analysis, thus making them useful for other developers. This classes have been inspired by the internal architecture of SAGA, overcoming some of its main limitations, such as the lack of flexibility in the design of graphical interfaces or the difficult combination of raster layers form different sources. While retaining its main advantages, the user experience has been improved. Most SAGA modules have been ported to this platform (those developed by the saga core team and also those developed by the Sextante team, which up to this date has been the main contributor to this project), excluding those already implemented somehow in gvSIG, such as file I/O modules. New modules have also been added, and others have been improved.

gvSIG has been developed with INSPIRE principles (Infrastructure for Spatial Information in the European Community) in mind. It Provides the most common GIS tools like data loading, map navigation, query map information like alphanumeric information, distance measurement, thematic cartography, legend edition using the most common legend types, labelling, feature selection by many selection types, data tables with statistics, ordering, table relations, table linking, layout manager, geoprocessing tools, CAD, raster processing, etc.
It is highly interoperable, i.e. it is able to work with most of the known data formats like ecw, ENVI hdr, ERDAS img, (Geo)TIFF, GRASS for raster; shapefile, GML, KML, DGN, DXF, DWG for vector, and databases like PostGIS, MySQL, Oracle, ArcSDE.
Its SDI client condition permits the connection to the OGC standards WMS, WFS (and WFS-T), WCS, and WMC.

Its functionality includes Geoprocessing tools like proximity (buffer, spatial join), overlay (clip, difference, intersect, union), advanced raster tools (georeferencing images, set image transparency, adjust bright and contrast, highlight, etc.) and advanced functionalities like Scripting support, a reprojection engine (PROJ4 wrapper), 3D visualization, network analysis, raster analysis features like classification or rectification, and gvSIG for mobile devices.

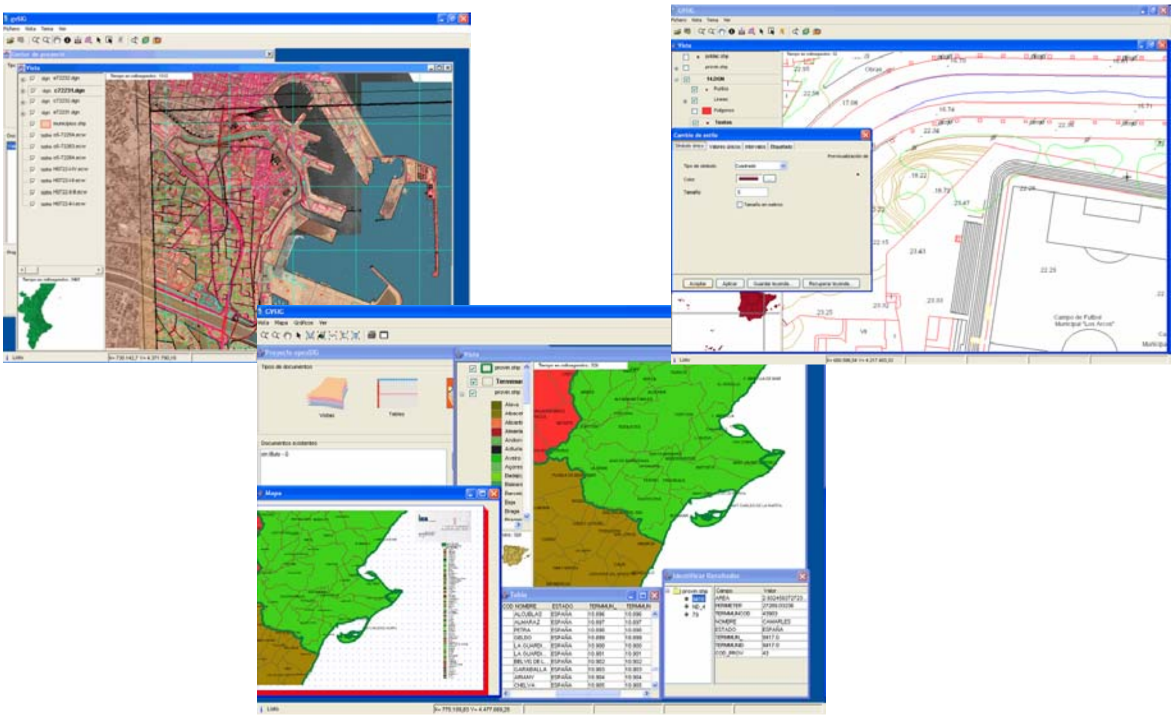


Figure 2: gvSIG user interface

Personal comments: gvSIG has an intuitive GUI which makes it easy to use. Still one can notice its Spanish origin in some missing translations. Via SEXTANTE there are a lot of spatial analysis tools available. The installation is a little tricky, as there are several packages with dependencies between each other. A very promising project!

3. Quantum GIS

Quantum GIS (QGIS) [6] is a user-friendly Desktop-GIS, released under the GNU General Public License (GPL) and available for Linux, Unix, MacOS X and Windows operating systems in several interface languages. Quantum GIS also acts as frontend user interface to the GRASS program since 2005 and thus benefits from the huge number of features of this package. The project itself started 2002.

Quantum GIS is implemented in C++ and includes Plug-In interfaces to extend the initial functionality. You can also use the software to configure UMN MapServer MAP-files, which reduces the effort needed to set up UMN MapServer map files.

Without the GRASS plug-in (and of course the installation of GRASS itself), QGIS mainly offers functionality as a GIS viewer and as a digitizing/editing tool. As GIS formats, it supports GDAL/OGR. Beside the GRASS extension, plug-ins mainly focus on simple vector layer processing (buffer, convex hull, intersection, difference, dissolve, union), geometry calculations, voronoi polygons, sampling, basic raster algebra. There exist also an extension for loosely coupling QGIS to R. It allows upload of QGIS layers directly into R, and the ability to perform R operations on the data directly from within QGIS.

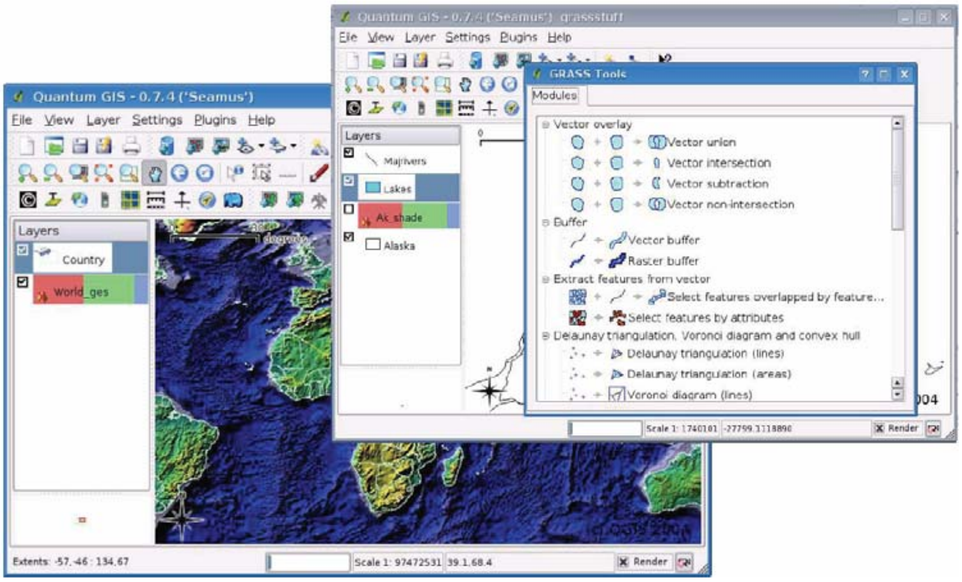


Figure 3: QGIS user interface

Personal comments: QGIS comes with a good documentation. The GUI is intuitive and easy to learn. Without GRASS its spatial analysis functionality is very limited. Good frontend for GRASS.

4. uDig

The User-friendly Desktop Internet GIS (uDig) [9] is a GIS application that serves as viewer and editor for geospatial data that is served via OGC compliant WMS and OGC WFS. uDig is implemented in Java and based on GeoTools.

uDig is not only used as an 'off the shelf product', but provides an environment to implement solutions for domain specific applications. The software has been developed by Refractions Inc., Canada and is available under the GNU Lesser General Public License (LGPL). Its main focus is on viewing and digitizing/editing GIS data.

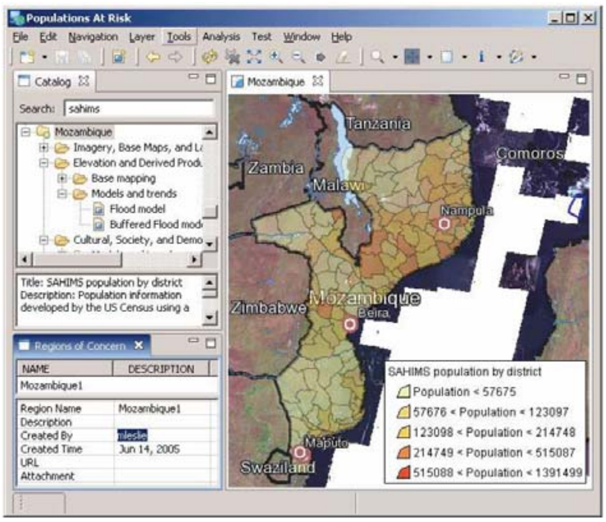


Figure 4: uDig user interface

uDig also serves as a base framework for other applications like forest management, hydrological modelling, route planning, etc.

5. SAGA

SAGA - short hand for "System for Automated Geoscientific Analyses" - is a free, hybrid, cross-platform GIS software [10]. The heart of SAGA is it's C++ and thus object oriented *Application Programming Interface* (API), providing data object definitions and computational methods for raster, vector and tabular data. As a normal user, you will not get into touch with the API. But as an interested scientist or coder you will soon discover it's great flexibility.

As user, you will most likely focus on using the steady growing availability of geoscientific methods. These are implemented in various SAGA modules which are bundled in so-called module libraries. These module libraries are accessible to you in different ways: By a *Graphical User Interface* (SAGA GUI) or by one of the scripting methods. The scripting methods include a *Command Line Interface* (SAGA CMD) which allows for batch/bash scripting (i.e. coupling of different modules to automate tasks) and a python interface which gives you also direct access to the SAGA API and is thus more flexible.

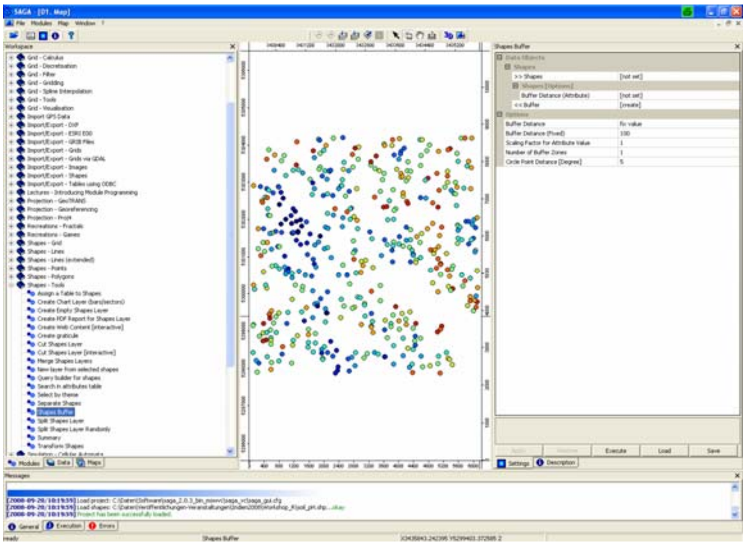


Figure 5: SAGA user interface

There are a lot of spatial analysis tools available, e.g.

- geostatistics (kriging, regression analysis)
- fuzzy logic
- grid analysis
- point pattern
- interpolation
- terrain analysis
- vector analysis

Personal comment: Still there is gap between the ambitious goal and its implementation. There are a lot of tools available, but most of them without any documentation. So it is sometimes hard to “guess” how to use the tools and what will be the result. The GUI is not very intuitive, as it is using a different approach than most GIS. The installation is very simple, as no registration is needed. Most of the tools have been translated to SEXTANTE, so using gvSIG gives an easier access to most of the tools.

6. GRASS

The Geographic Resources Analysis Support System (GRASS) [11] is the most complex and professional Open Source based desktop GIS and is supplied under the GNU General Public License (GPL). Supported platforms include Linux, Unix, MacOS X and Windows. The strength of this GIS lies in its modular structure and its utility for raster processing and modelling.

Users can interface with the software features through a graphical user interface (GUI) by or by "plugging into" GRASS via Quantum GIS. They can also interface with the modules directly through the modified version of the shell that the application launches or by calling individual modules directly from a preferred shell (this latter method requires the setting of several environment variables).

The recent GRASS 6 release introduces a new topological 2D/3D vector engine and support for vector network analysis. Attributes are managed in .dbf files or SQL-based DBMS such as MySQL, PostgreSQL/PostGIS, and SQLite. The system is capable of visualizing 3D vector graphics data and voxel volumes. GRASS supports an extensive range of raster and vector formats through the binding to GDAL/OGR libraries, including OGC-conformal (Open Geospatial Consortium) Simple Features for interoperability with other GIS. It also supports Linear Reference System.

The GRASS Development Team is a multi-national group consisting of developers at numerous locations. GRASS is one of the eight initial Software Projects of the Open Source Geospatial Foundation.

GRASS is designed as an environment in which tools that perform specific GIS computations are executed. Unlike typical application software, upon starting GRASS, the user is presented with a UNIX shell containing a modified environment that supports the execution of GRASS commands (known as modules). The environment has a state that includes such parameters as the geographic region covered and the map projection in use. All GRASS modules read this state and additionally are given specific parameters (such as input and output maps, or values to use in a computation) when executed. The majority of GRASS modules and capabilities can be operated via a graphical user interface (provided by a GRASS module), as an alternative to manipulating geographic data in shell. There are over 200 core GRASS modules included in the GRASS distribution, and over 100 add-on modules created by users and offered on the GRASS web site. The GRASS libraries and core modules are written in C; other modules are written in C, UNIX shell, Tcl, or other scripting languages. The GRASS modules are designed under the Unix philosophy and hence can be combined using shell scripting to create more complex or specialized modules by a user without knowledge of C programming.

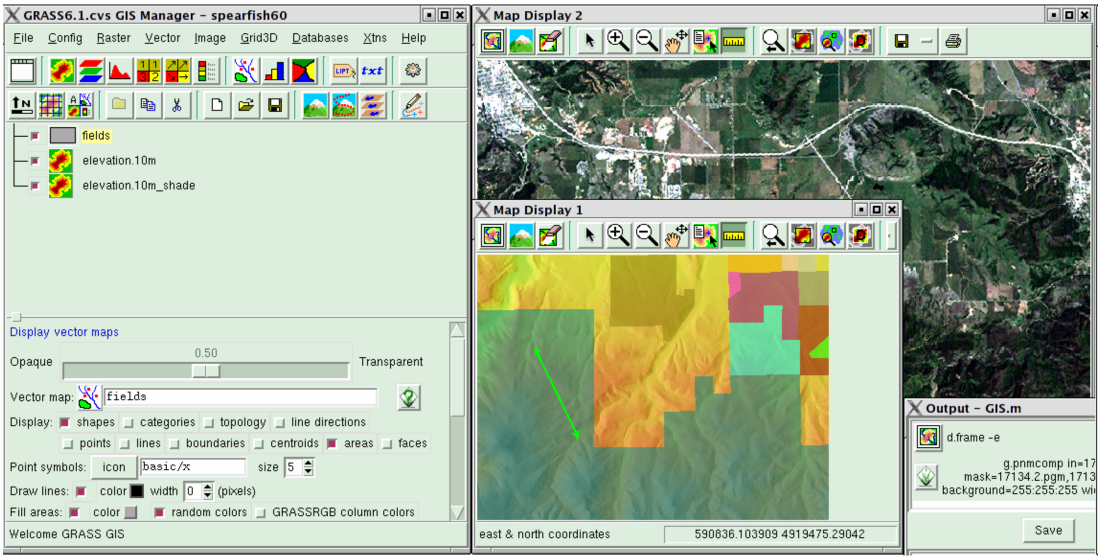


Figure 6: GRASS user inteface

7. ILWIS Open

The Integrated Land and Water Information System (ILWIS) is a PC-based GIS & Remote Sensing software, developed by ITC (International Institute for Ge o-Information Science and Earth Observation, The Netherlands) from 1988 up to its last release (version 3.3) in 2005. ILWIS comprises a complete package of image processing, spatial analysis and digital mapping. It is easy to learn and use; it has full on-line help, extensive tutorials for direct use in courses and 25 case studies

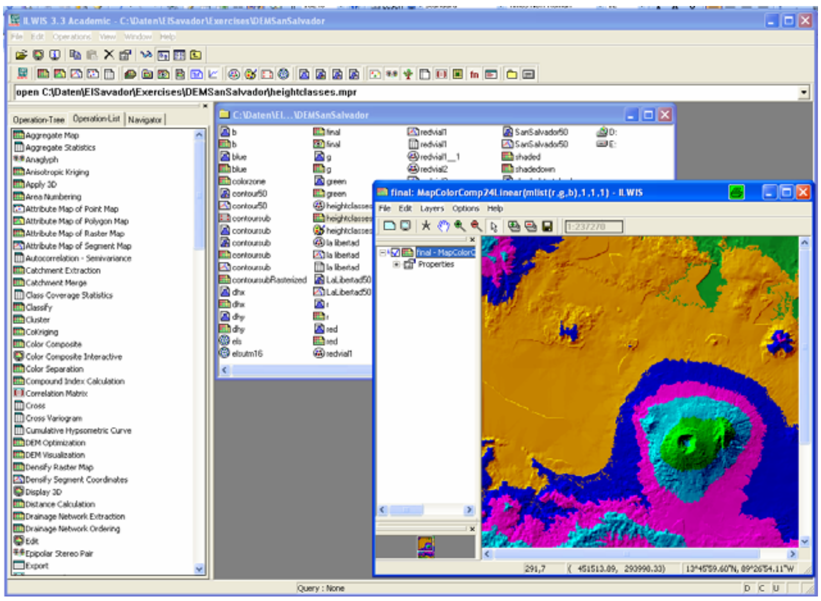


Figure 7: ILWIS user interface

of various disciplines. It has been originally programmed in C, the open source version has switched to an MS Visual Studio (,NET) project.

As per July 1st, 2007, ILWIS software is freely available ('as-is' and free of charge) as open source software (binaries and source code) under the 52°North initiative (GPL license). This software version is called **ILWIS 3.4/3.5 Open** [12].
Some of the key features are:

- Integrated raster and vector design
- Import and export of widely used data formats
- On-screen and tablet digitizing
- Comprehensive set of image processing tools
- Orthophoto, image georeferencing, transformation and mosaicing
- Advanced modeling and spatial data analysis
- 3D visualization with interactive editing for optimal view findings
- Rich projection and coordinate system library
- Geo-statisitical analyses, with Kriging for improved interpolation
- Production and visualization of stereo image pairs
- Spatial Multiple Criteria Evaluation
- WMS (new in V3.5)
- Surface Energy Balances (new in V3.5)

Personal Comment: ILWIS is still focussing more on raster than on vector formats. The difference between ILWIS and other GIS is its outstanding documentation from ITC. There exists not only a very good manual, but many tutorials showing the application of ILWIS in many different fields. Even if you are using a different GIS, it is sometimes worthwhile to have a look into these tutorials.

8. Outlook and Discussion

It looks like a big diversity of applications. But, except the two “dinosaurian” GRASS and ILWIS, which are on the one hand based on a rather old technology, on the other hand mature and well documented, there are many links between the new projects, all emerged in the last 5 to 7 years.
Concerning the supported data formats, most of them are based on the same libraries: OGR [13] for vector data and GDAL [14] for raster format. Both libraries can also be used as a toolkit without a GIS for transformation between different data formats. For coordinate systems and re-projections, most of the OS GIS are using the library proj4 [15].

The main difference between the developments is mainly the programming language used. Still it is argued that Java programs are not as fast as C++ programs working on big datasets. As operating platform, LINUX is loosing some importance; the use of Windows is increasing. MacOS users form a much smaller, but stable group.

To synchronize and promote the development of OpenSource GIS the OSGeo (Open Source Geospatial Foundation) has been founded. The Open Source Geospatial Foundation has been created to support and build the highest-quality open source geospatial software. The foundation's goal is to encourage the use and collaborative development of community-led projects. For desktop GIS, OSGeo promotes GRASS, QGIS and gvSIG as projects.

As for many other Open Source project one of the major drawbacks is the lack of a good documentation. Free software development is driven by the (developer) needs, not really by the market. The demands fixed in a road-map for further development seems sometimes not very realistic. The focus on further development described inn most of the road-maps is to integrate more geoprocessing functionality and to use the software as elements of a geospatial infrastructure based on the OGC specifications.

Overview of free and open source desktop GIS

		language/ technology	OS	licence	community	dissemination	features	Application focus
jump (Java Unified Mapping Platform) 2002	gvsig** (Generalitat Valencia Sistema de Información Geográfica) 2004	Java	Windows, Mac, Linux, …	GPL	small group of developers, many plug-ins developer, many user	Spain/Europe	WMS, WFS-T, GML3, digitizing/editing, analysis (vector), many plug-ins (SEXTANTE),	Viewing, analysis
	deeJump				small group of developers, few users	Germany/Europe	WMS, WFS	OGC viewer for WMS, WFS
	openJump				small group of developers, few users	Europe/America	WMS, WFS, PostGIS, digitizing/editing, plug-ins	Viewing, editing
	Kosmo 2006				small group of developers, few users	Spain/Europe	under development	Corporate GIS Platform (desktop, server, web clients, PDA clients,…)
uDig (user-friendly Desktop GIS 2004		Java, Eclipse	Windows, Mac, Linux, …	LGPL	small group of developers, but active many users	North America	visualization, WMS, WFS-T, GML3, Shape, PostGIS, digitizing/editing, (GeoTools)	Viewing, editing
Quantum GIS* (2002)		C++, Qt, Python API	Windows, Mac, Linux, …	GPL	about 20 core developers, community large and active	worldwide	WMS, WFS-T, OGR-vector formats, GDAL-raster formats, PostGIS, many plug-ins (vector analysis)	Viewing, GRASS-Graphical User Interface
SAGA (System for Automated GeoScientific Analysis) (2001)		C++	Windows, Linux	GPL/LGPL (GUI)	small, academic	Germany/Europe	GDAL-raster, Spatial Analysis (hydrology, kriging, fuzzy, filter, simulation)	Analysis, modeling, scientific visualization
GRASS* Geographic Resources Analysis Support System (1982)		C	Windows, Linux, Mac	GPL	large dev, many users	worldwide	WMS, WFS, OGR-vector	Analysis and scientific visualization, cartography, simulation
ILWIS (1988/2007)		Visual C/ .NET	Windows	GPL	small dev, many users	worldwide	Raster analysis	Analysis

*osGEo project
* in incubation process for osGEO project

SEXTANTE is based on SAGA tools

9. Links & references:

[1] http://thuban.intevation.de
[2] http://www.mezogis.org
[3] http://openjump.org
[4] http://www.gvsig.gva.es
[5] http://www.sextantegis.com/en/index.htm#
[6] http://www.qgis.org/
[7] http://www.deejump.com/
[8] http://www.opengis.es/
[9] http://udig.refractions.net
[10] http://saga-gis.wiki.sourceforge.net/
[11] http://grass.osgeo.org/
[12] http://52north.org/index.php?option=com_content&task=view&id=131&Itemid=319
[13] http://www.gdal.org/ogr/
[14] http://www.gdal.org/
[15] http://trac.osgeo.org/proj/

Paul Ramsey: The state of Open Source GIS. Sribd, 2007
http://www.scribd.com/doc/510902/Survey-on-Open-Source-GIS

Gary E. Sherman: **Desktop GIS. Mapping the Planet with Open Source Tools**
The Pragmatic Programmers. First Edition Oktober 2008

Part B: Case Study with gvSIG

1. The case study Schluchsee

In the following, a short introduction into the free and Open Source desktop GIS gvSIG will be given. For this introduction, a case study will be used. For the region of the Schluchsee, Germany, the damage of trees due to air pollution has been surveyed. About 36 000 trees have been classified in 4 groups (1= no damage, 2=small damage, 3=medium damage, 4=high damage). The effect of the terrain should be investigated on the degree of damage. Has the elevation of the terrain any correlation with the degree of damage? As the winds in this region are mainly blowing from the west, it is imagined that there will be also some correlation with the aspect of the terrain, i.e. the exposition of the terrain towards north. Is this assumption reflected in the data?

2. The gvSIG user interface

The user interface of gvSIG reminds of "old" ArcView, anybody who has worked with ArcView will soon feel familiar with it. The purpose of the *Project Manager* is to manage the documents of a gvSIG project, i.e. views, tables, and maps. The project settings are store in a *.gvp file, *.gvt template files are available for different map layouts.

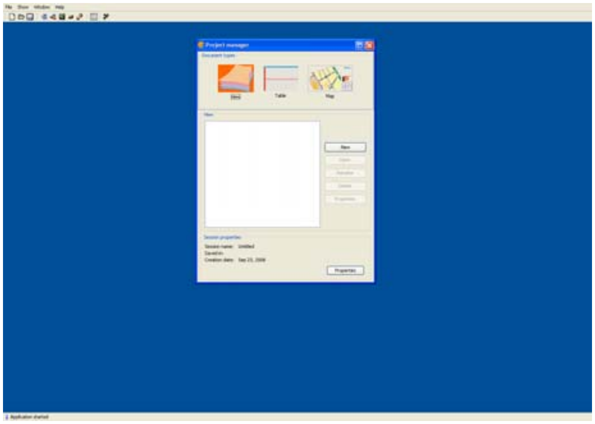


Figure 8: The gvSIG GUI after starting

Before loading the data, a good idea is to do some general settings (*preferences*). Here the folder locations for the project and the data should be set, as well as the coordinate reference system used for our project. As all our data is with respect to the Gauss-Krüger map projection (zone 3), the EPSG number 31467 should be used.

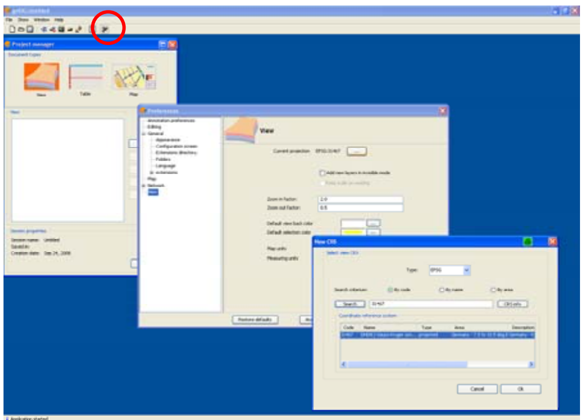


Figure 9: Setting project preferences

Our data for an analysis will be managed in a view. So next we should create a new view, renaming it for instance to *Schluchsee_elevation*.

3. Loading the data

After opening the view, the data can be loaded. There are two data sets available:

- *trees.csv*, a semicolon separated text file with locations and damage classification of the trees
- *dem.asc*, a text file in the ESRI-grid format with elevation data in 25m resolution

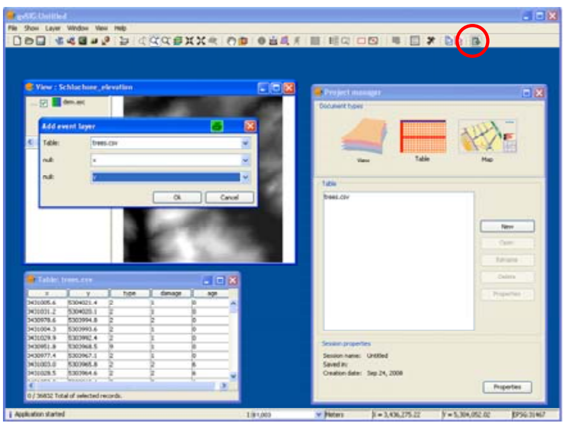


Figure 10: Loading the table tree file as event layer

The DEM can directly be loaded using the *Add layer* button for the active view. From the *Add Layer* dialog use the tab *File*, click

Add and change in the following dialog the *Files of Type* setting to *gvSIG Image driver*. Now the *dem.asc* file can be selected and loaded.

As the tree data is up to now no point data set but contains the coordinates as columns, we have first to add the file as a *table* and then add the data set as an *event theme* to the view. The x any column have to specified as coordinate fields.

After loading the data in the *property* dialog for the two layers (right mouse click on the layer entry in the table of content) the symbology (colour, size, etc.) can be changed.

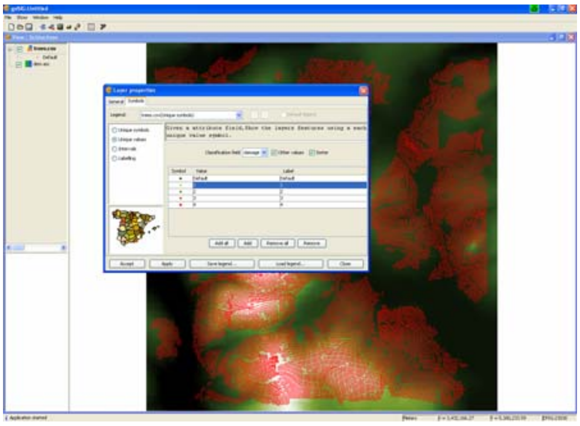


Figure 11: Changing the default symbology

4. Analysis

For the analysis of the data, we will calculate how many damaged trees are in each (normalized) elevation class. The normalization is necessary, as the area covered by the different elevation classes will be different. So the formula we will use is quite simple:

$$I_{idamage} = \frac{n_i / a_i}{\sum n_i / \sum a_i}$$

The damage index will be calculated by dividing the number of damaged trees for each class by the area of this class, normalized by the quotient of the total number of damaged trees and the total area of our study area. If the index is 1 for one class, we would have exactly an average number of trees, if the index is smaller than 1, the number is smaller than the average, if the index is bigger than 1, we have more damaged trees than the average of all classes.

To calculate the index, the following analysis steps have to be performed:
- creating elevation classes
- calculating the area for these classes
- selecting only the trees with medium and high damage
- counting the damaged trees in each elevation class
- calculating the damage index for each elevation class

4.1. Creating elevation classes

For creating the elevation classes we have to reclassify our DEM. As the DEM is a raster, we have to use the tools from the SEXTANTE toolbox.
In *Basic tools for raster layer* we will first use the tool *Basic Statistics* to get an idea of the elevation in our study area.
From the *max* and *min* value we can see that the height difference in our study area is about 450m, so with 9 classes we would have a height classes each 50m.
For the reclassification, in *Tools for categorical raster layers* the tool *Divide in n classes of equal amplitude* can be used. The reclassified DEM is still a raster layer. For counting the trees, we have to convert the different height classes into vector polygons (vectorization).

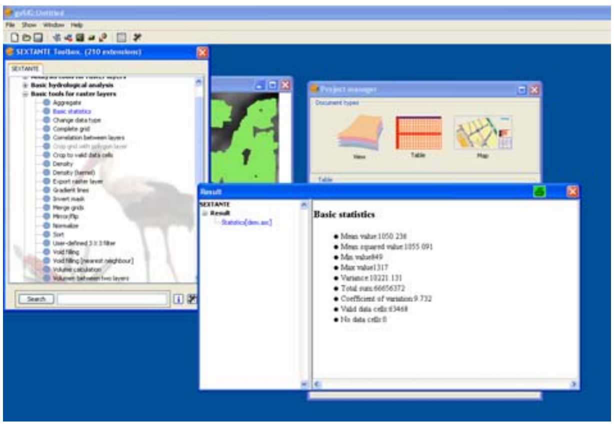


Figure 12: SEXTANTE basic raster statistics

From the resulting polygon vector layer and the associated table (select the layer in the table of contents and click the *Show attributes of selected layer* button to open the attribute table) we can clearly see that there are 9 elevation classes but 61 polygons. What we need for our analysis is one object (maybe with several polygons) for each class, so we can count the trees falling in each class.

So we have to dissolve the polygons, i.e. merging all polygons belonging to the same class to one object. This can be achieved using the *Geoprocessing Tool box* of gvSIG (*select View -> Geoprocess Toolbox -> Geoprocess -> Aggregation -> Dissolve*). As all our polygons have the field *dem.[re* (which can be of course renamed, to give it a more meaningful name, see below), we can use this field as dissolve attribute. The new layer contains now only 9 objects, one for each class.

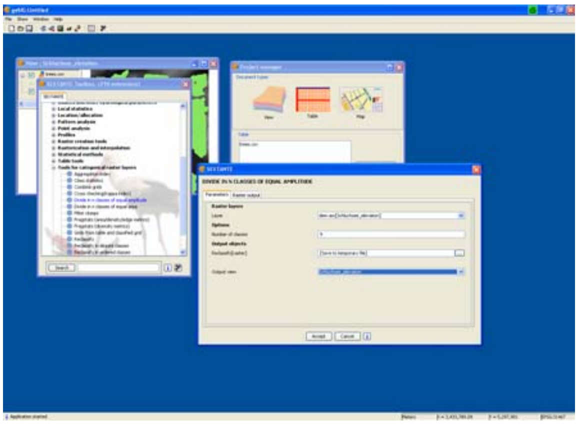


Figure 14: Reclassifying the DEM into 9 classes

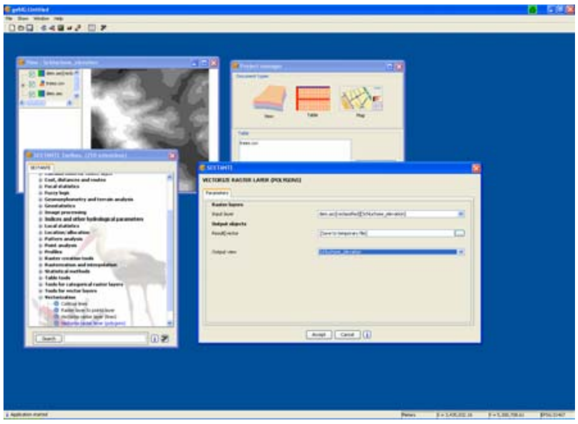


Figure 15: Vectorizing the polygons

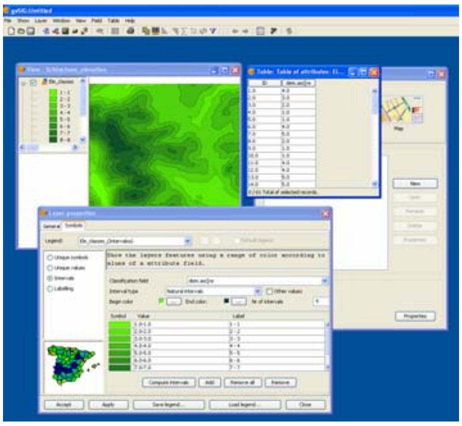
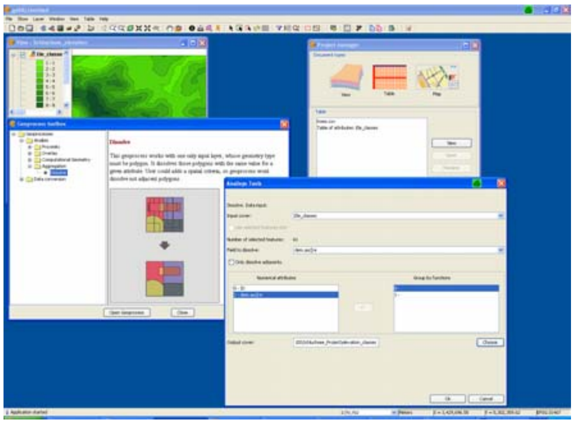


Figure 16: 61 polygons for 9 classes



Dissolving the polygons

4.2. Selecting the tree with damage

We are only interested in trees with medium and high damage. So we will first select them, using the *Filter* tool. From the attribute table it can be clearly seen that 12272 trees out of 36832 have been selected.
To continue our analysis, we need the filtered trees as a separate layer, so we will export them to a new shape-file.

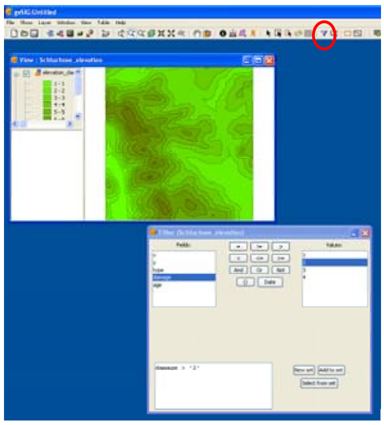


Figure 17: Filtering damaged trees

Select from the main menu *Layer -> Export to -> shp…* and confirm the number of the selected features. Specify a location for new file and add the layer to your view.

4.3. Counting the trees

Now we need to know the number of trees in each elevation class. From the SEXTANTE tool box we find *Tools for vector layers -> Count points in polygons*. Opening the attribute table of the new layer we will see the number of trees in each elevation class.

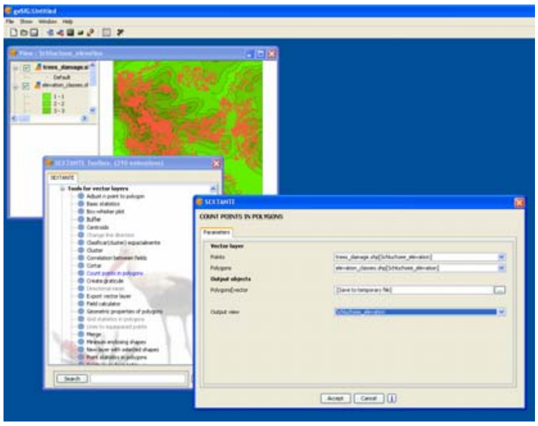


Figure 18: Counting points and polygons

4.4. Calculating the damage index

In the last step we will calculate the damage index. First we need to know the area for each of our elevation classes. This calculation can be done again with SEXTANTE tools: *Tools for vector layer -> Geometric properties of polygons*. A new layer will be created containing additional fields like area, perimeter, etc.
Now we can calculate the damage index, we have all the information we need. First we have to add a new field to the attribute table, and then we can use the field calculator to enter our formula. The total number of damaged trees we know already (which has to be readjusted, as some are lying on the border of two classes and are counted twice), the total area of the study area is needed too. Both numbers we can get from the *statistics* of the appropriate columns.

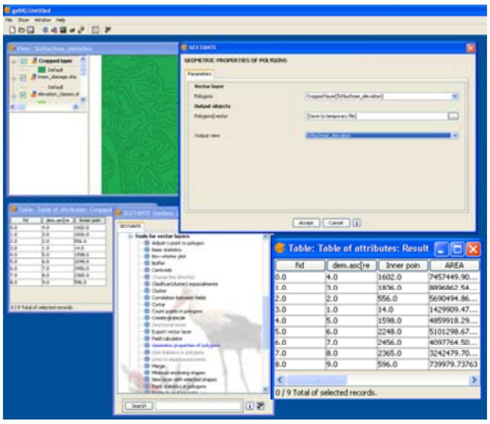


Figure 19: Calculating the area

For adding a new field, we have first to switch to the *editing* mode. Select the layer to edit from the table of content of the view and click on *Layer -> Start edition*, the layer name will change its color to red.

In the edit mode, we can also change the structure of a table by deleting and adding fields, renaming fields etc. using the table manager (open the attribute table and click on *table -> Manage fields*). We have to add at least one new field of type *double* for our damage index.

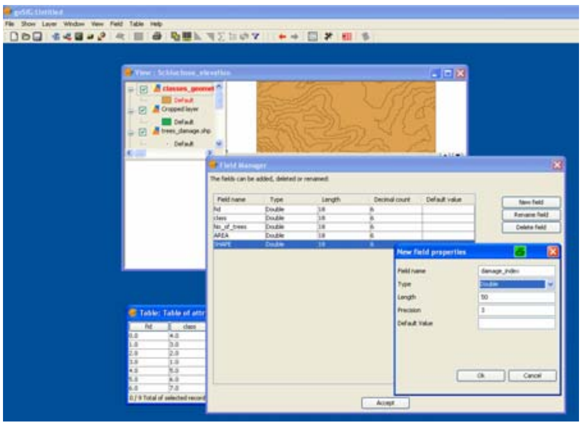


Figure 20: Managing fields

After accepting our changes we are ready for calculating the new field values. Select the new field in the table and use the *Expression* button from the main menu to enter the formula: *([No_of_trees]/[AREA])/(13271/4.1516E7)*

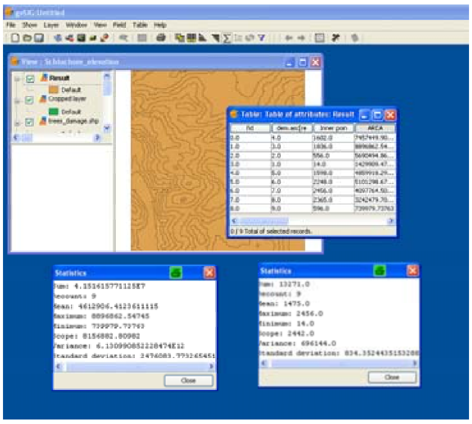


Figure 14: Statistics for total numbers

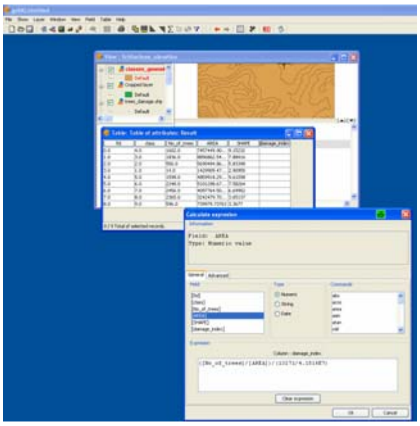


Figure 15: Entering the formula for the new field

From the final result we can clearly seen that there is a strong correlation of damage and height: the higher the elevation the more damage we have.

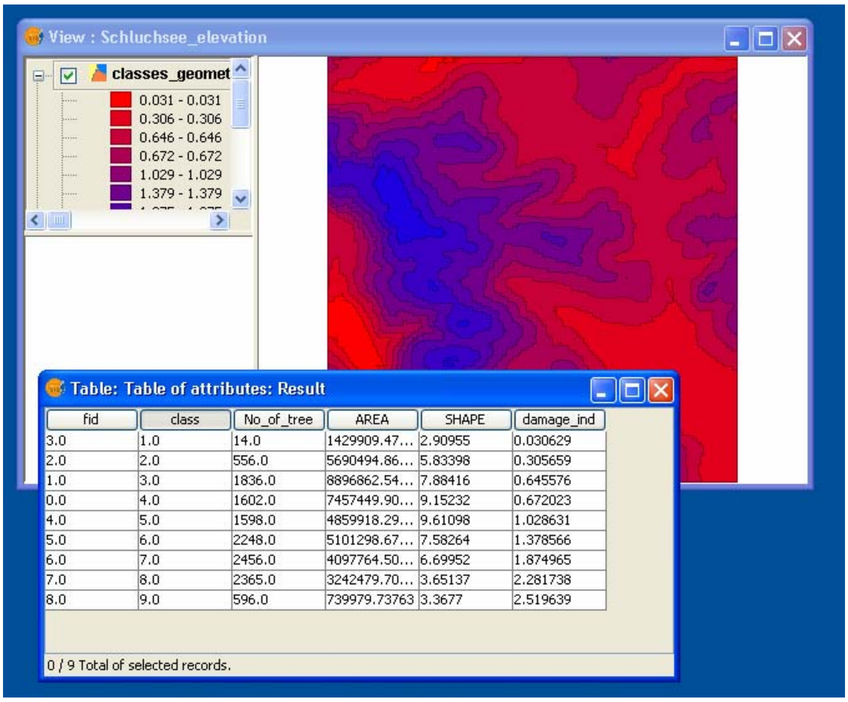


Figure 21: result: the damage index of height classes

4.5. Designing the Final Mal Layout

In the Map view the final map layout for printing can be designed. The map layout can be exported to PDF or Postscript.

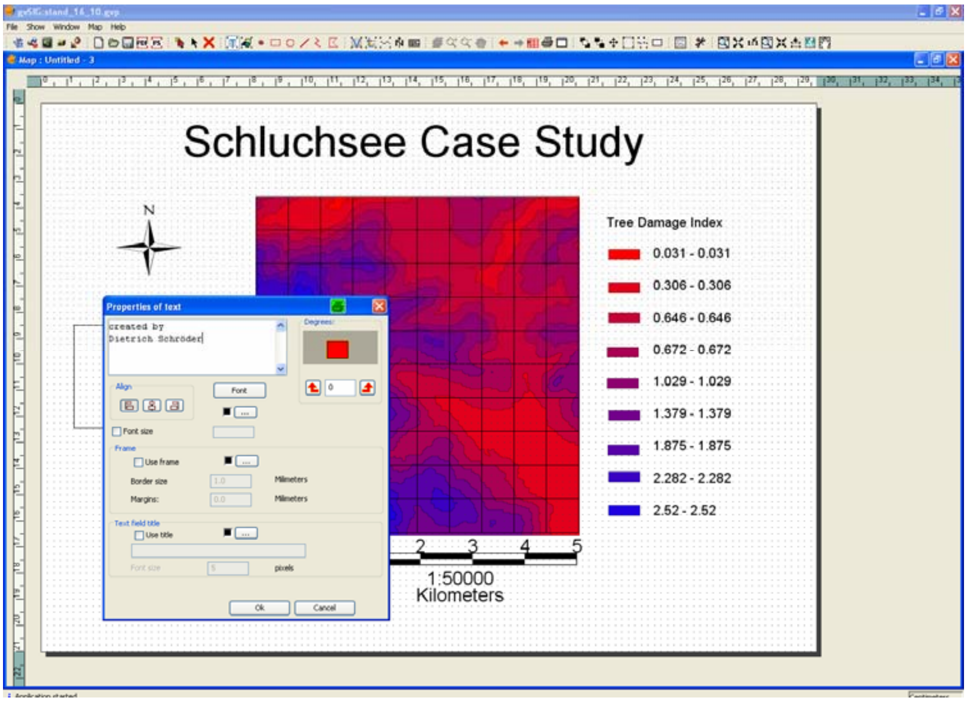


Figure 22: Designing the final map layout

5. Exercise: Analysis of the aspect

Task: Do a similar analysis for the aspect!
Remark: The aspect will be calculated in radian, 0 pointing north. Make sure to find appropriate aspect classes for your analysis!