ARTICLE IN PRESS

Computers, Environment and Urban Systems xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Computers, Environment and Urban Systems

journal homepage: www.elsevier.com/locate/compenvurbsys

Review

The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption

Stefan Steiniger *, Andrew J.S. Hunter

Department of Geomatics Engineering, University of Calgary, 2500 University Drive N.W., Calgary, Alberta, Canada T2N 1N4

ARTICLE INFO

Article history:
Received 17 January 2012
Received in revised form 1 October 2012
Accepted 7 October 2012
Available online xxxx

Keywords:
GIS software
FOSS4G
Free software
Open source
Overview
Software selection

ABSTRACT

Over the last decade an increasing number of free and open source software projects have been founded that concentrate on developing several types of software for geographic data collection, storage, analysis and visualization. We first identify the drivers of such software projects and identify different types of geographic information software, e.g. desktop GIS, remote sensing software, server GIS etc. We then list the major projects for each software category. Afterwards we discuss the points that should be considered if free and open source software is to be selected for use in business and research, such as software functionality, license types and their restrictions, developer and user community characteristics, etc. Finally possible future developments are addressed.

Contents

* Corresponding author. Tel.: +1 403 2206995; fax: +1 403 2841980.
E-mail address: ssteinig@ucalgary.ca (S. Steiniger).

0198-9715/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

1. Introduction

Several software projects that focus on the development of free and open source Geographic Information Systems (GIS) software have been initiated in the last 10 years. Although there have only been a few new projects since the last general reviews by Ramsey (2006, 2007), changes in the software user and developer communities have led to a different "market" environment than that which existed 5 years ago. In particular, the Quantum GIS project, the PostGIS project, and the OpenLayers project have been able to attract users and developers, which has subsequently influenced software functionality and support and as a consequence raised their attractiveness to new users of free software.

Our goal is to present an overview of (major) GIS software projects that develop free and open source software in 2012 to (i) facilitate free and open source GIS adoption in business, research and teaching, and (ii) to further its utilization for development and research projects. Hence, this article does not focus on the application of free GIS software for research and industry, nor will we discuss the possible impact of Free and Open Source Software (FOSS) on research. We shall first discuss what free and open source software is. Then we identify the foundations of various software projects and arrange them within functional categories of GIS software such that a comparative analysis may be undertaken. A short guide on the selection of free software is given including criteria that should be considered prior to its adoption. An overview on further resources, i.e. books, articles and webpages, are presented before we conclude by looking into the possible future. We note that we will provide the web links to software project web pages in an article Supplement.

2. 'Free Software' – What is it about?

Some may believe that 'free' software exists because a group of people must have been really unhappy with the price they had to pay for software. That was, however, not the reason why the Free Software Foundation (FSF) was created and 'copy-left' license types emerged. Rather, a group of people, and in particular Richard Stallman, have been unhappy in the 1970s and 1980s when software and hardware companies (i) started to unbundle hardware and software, and (ii) introduced restrictions to the access of source code for hardware drivers and software, after changes in US patent laws (Grassmuck, 2004). These people were unhappy, because the access restrictions prohibited them to see how other programmers controlled hardware (e.g., printers or monitors), nor could they change the code to improve it (as it was now pre-compiled), nor could they send anymore changes (i.e. 'hacks') to friends to get their opinion, or even to simply help them with a computer problem (Stallman, 1999; Grassmuck, 2004).

With this historical perspective it becomes clear that the idea of 'free software' has its origin in the idea of freedom (as in free speech) and not in the idea of free-of-cost. This also includes that a distinction between 'free software' and 'commercial software' is neither correct nor expresses the thinking of the creators of free software. Rather, a distinction should be made between 'free software', that grants freedoms of use, modification and re-distribution to the public, and 'proprietary software', that takes these freedoms away (in most cases at least). To distinguish between free and proprietary software the reader should study the licenses that come with each type of software. He will discover that 'free software' grants the following four freedoms: (1) the freedom to run the software for any purpose (e.g., may it be education or business), (2) the freedom to study and adapt the software for own needs, (3) the freedom to redistribute the software, and (4) the freedom to improve the software and to release improvements to the public.[1]

When studying the licence terms further, for instance those of the GNU General Public Licence (GNU GPL), it can also be obtained that there exist no license term that prohibits selling the software to others. Hence, selling 'free software' is possible and is, in-fact, done by companies and programmers to make a living (see below). Also, from reading the freedoms it becomes clear that access to source code is a requirement for being able to grant freedoms (2) and (4). With that, access to source code cannot be seen as a 'bonus'.

After we have been referring in the previous paragraphs only to 'free software', the reader may wonder why throughout the article we use the term 'Free and Open Source Software (FOSS)'. One reason is that with using the term 'free software' we provoke confusion with the meaning of 'free-of-cost'. However, also the term 'open source software' that is used by some is not sufficient, when referring in-fact to 'free software'. Because the attribute 'open source' indicates only that the software's source code is accessible, but it does not say if the source code can be modified and if it can be given to someone else (Stallman, 2009). Hence, the combination of terms into one: 'Free and Open Source Software', i.e. FOSS, seems best. A discussion of the different types of software licenses can be found in Johnson (1999), on the FSF web pages (FSF, 2012a, 2012b), and in Steiniger and Bocher (2009) among many other web resources. For the reminder of the article we use the abbreviation 'FOSS4G' – "Free and Open Source Software for GIS/Geospatial' to encompass free software related to the GIS domain. However, FOSS4G also denotes a conference series on free GIS software and activities.

3. Founding FOSS4G projects

Who are the people that use, develop, and finance free software projects, and why? This is probably one of the more commonly asked questions by people taking their first steps into the FOSS4G world. The survey by Steiniger and Bocher (2009), while restricted to desktop GIS systems, shows that free GIS projects are founded and maintained by (i) companies, (ii) individuals (e.g., freelance programmers or GIS specialists), and (iii) institutions, such as research institutes, universities, and public authorities. The motivation driving these actors is generally quite different (see Wheeler, 2007):

Companies may be hired by public institutions to develop software, but may also show initiative on their own, if they see that some (niche) tool/software to accomplish certain (GIS) tasks is missing. The non-existence of a certain tool is typically due to two factors: (i) the market is too small to generate an 'interesting' profit for a software provider to develop a specialized tool, or (ii) it is a new innovative product that solves new types of tasks that have emerged from technological advances. But what business case drives a company to release new software developments under a free software license? Generally, business cases are based on services. That is, companies generate income by offering training courses, selling support material, providing tailored solutions based on free software, providing maintenance and (hot-line) support, developing customized functions, etc. (see also Christl, 2008; Ramsey, 2009). Thus, free GIS software companies generate their income in a similar manner to GIS

[1] see http://www.gnu.org/philosophy/free-sw.html.

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

Table 1
Typical tasks accomplished with different GIS software.

GIS task vs. GIS software	Query/select	Storage	Exploration	Create maps	Editing	Analysis	Transformation	Creation	Conflation
Desktop GIS									
– Viewer	●	●	●	○					
– Editor	●	●	●	●	●		○	●	
– Analyst/Pro	●	●	●	●	●	●	●	●	●
Remote sensing software		●	●	○	●	●	●		
Explorative data analysis tools	●	●	●	●	○	●	●		
Spatial DBMS	●	●				○	●		
Web map server	●		●	●	○			○	
Server GIS/WPS server	●	●		●		●	●		●
WebGIS Client									
– Thin client	●		●						
– Thick client	●	●	●	●	●	●		●	
Mobile GIS	●	●	●		●			●	
GIS libraries		●		●		●	●		●

● – Standard functionality; ○ – Optional functionality.

consulting companies, as opposed to typical software companies that rely on selling software licenses.

Reasons why companies publish their developments under a free software license are manifold. First, free licenses provide an easier way to collaborate on software development with other companies or institutions (Wheeler, 2007). Sharing development allows organizations to combine different (expert) resources on the one hand and reduce development costs on the other (Perens, 2005). Opening code to the public can also be attractive to others and other companies, individuals, or institutions may choose to assist with software development. This is attractive because contributing to a project in this manner may provide others an opportunity to address their specific software needs. The contrasting options would be to either start a completely new (proprietary) software project on their own – at a comparably high cost -, or to integrate the special functions that are needed into proprietary software. However, the second option has the disadvantages that it (i) binds the custom function to a (non-free) software product, i.e. introduces dependencies, and (ii) a full integration of the customized function in proprietary software by the software vendor is rather rare, as it depends on the size of the target user community (see also discussions by Steiniger and Hay (2009) and Rey (2009)). The ability to be able to truly influence software development and functionality to fit ones particular needs is one of the major reasons for public authorities to fund free GIS software development (see for instance the desktop GIS projects JUMP, uDig and gvSIG). Another major reason for authorities is the ability to save costs due to free in-house distribution of software, i.e., no restrictions on the number of software copies that can be run at the same time.

An example for initial funding of a FOSS project by a research agency is the MapServer project that was co-funded by the US National Aeronautics and Space Administration (NASA). The initiation and development of free GIS software projects by research institutions is usually driven by the need to develop new methods for data analysis, visualization, distribution, etc. This was the case with the MapServer project that aimed for new methods of data visualization and distribution over the Internet. Research projects are typically constrained by funding, making it difficult to budget for software licenses, and at the same time have the technical capability to customize software to the needs of the research. Hence, the FOSS model provides a good alternative to research with proprietary software. The ability to freely distribute and collaborate on software development is additionally attractive (Steiniger & Bocher, 2009; Steiniger & Hay, 2009), and essential in terms of peer reviewing the underlying methods (Rocchini & Neteler, 2012). Research investments in FOSS can be considered less risky because if a project is not continued, then other or future projects can continue as the source code remains for usage elsewhere.

The motivation for an individual to contribute, e.g. by answering questions, writing documentation, develop software, etc., in FOSS projects can be very diverse. Some see such projects as an opportunity to start their own private consulting business, whereas others contribute for more intrinsic reasons, i.e., they may be hobbyists with a desire to learn, or because they receive recognition from software users and other developers (Finck & Bleek, 2006; Hars & Ou, 2002). However, as Finck & Bleek, (2006) point out, the main argument for an individual's participation in FOSS projects is typically a personal benefit.

4. Categories of GIS software

Software that is used to create, manage, analyze and visualize geographic data is usually incorporated under the umbrella term 'GIS software' (Steiniger & Weibel, 2009). Different functional categories of GIS software can be identified with respect to the tool sets that GIS software offers, and with respect to the tasks that can be accomplished. Such tasks involve basic activities that are common to daily computer usage such as (1) data visualization and exploration, (2) data creation, (3) data editing and (4) data storage. Other common GIS tasks include (5) data conflation, i.e., integration of data from different sources (Blasby, Davis, Kim, & Ramsey, 2002), (6) data queries to select a subset of the data, (7) data analysis, which we consider to be the creation of new information (output) from existing data (input), (8) data transformation, as some analysis tasks require the user to transform, or manipulate, the data beforehand (e.g. transform the data into a different coordinate system, or convert them from raster to vector format), and lastly, (9) the creation of maps – the most common method used to visualize analysis and query results.

Steiniger and Weibel (2009) identified seven major types of GIS software: (i) Desktop GIS, (ii) Spatial Data Base Management Systems (SDBMS), (iii) Web Map Server, (iv) Server GIS, (v) Web GIS clients, (vi) Mobile GIS, and (vii) Libraries and Extensions. For the purpose of this survey we did split the last category "Libraries and Extensions" into the two categories: "Libraries" and "GIS Extensions, Plug-ins and APIs". Web Map Server and Web GIS clients have been subsumed under the category "Software for Internet Mapping Applications". Furthermore, we extend the set to include two additional categories (viii) Remote Sensing Software, which could be considered a special form of desktop GIS, and (ix) Exploratory Spatial Data Analysis (ESDA) software (Anselin, Syabri,

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

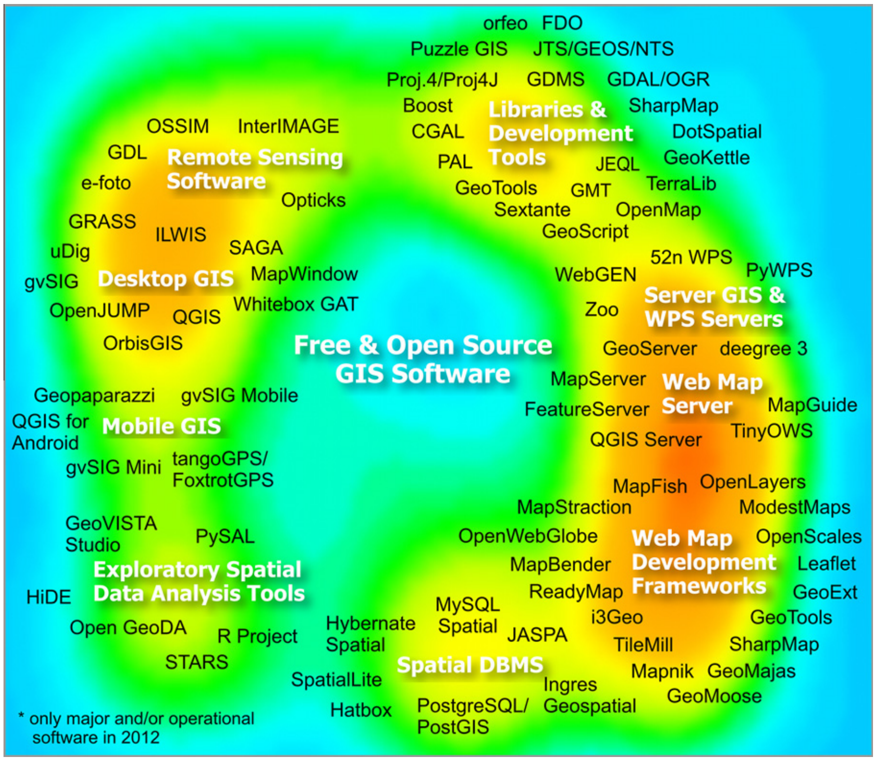

Fig. 1. The free and open source geographic information software map of 2012.

& Kho, 2006). Table 1 characterizes the different software types with respect to GIS functionality as defined above.

5. An overview of (major) GIS software projects

In this section we give an overview of FOSS4G software projects with respect to the different functional GIS software categories. We have restricted the overview to established projects that continue to maintain an active community, and new projects that appear promising given their ability to rapidly attract a strong following of users and developers. We have also attempted to provide a general assessment of the state of each GIS software category. Fig. 1 presents the project overview in a spatialized form.

5.1. Desktop GIS

Desktop GIS software is probably the most common GIS software in use. The company ESRI – Environmental Systems Research Institute (2012) defines desktop GIS as "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." Steiniger and Bocher (2009) add: ''That is, the software is not executed on a server and remotely accessed or controlled from or by a different computer.'' Hence, all traditional GIS tasks, i.e., not tasks related to web and remote processing applications, can be accomplished with a desktop GIS (see also Table 1). Sometimes, proprietary GIS software vendors distinguish in software pricing between two or three categories of desktop GIS with respect to the functionality that the software offers. In the case of 3 desktop GIS categories so-called viewer applications offer functionality for viewing and exploring data, while editor applications provide, in addition to viewer capabilities, the ability to create and update spatial data. Analysis GIS software, also termed ''Professional edition'', typically offers the highest level of functionality by adding functions for data analysis, map creation, data conflation, etc., to editor applications.

Due to the ''work horse'' status of desktop GIS, i.e., the multitude of tasks that can be accomplished with it, there are a large number of free software projects that develop and maintain FOSS desktop GIS. Recent reviews by Steiniger and Bocher (2009) and Steiniger and Hay (2009) identify eight mature desktop GIS projects. A project is considered to be *mature* if (i) functionality is comparable to proprietary GIS software (at the viewer level), (ii) software support is available through commercial service providers, and (iii) the user and developer communities have an international focus. The eight[2] projects are: (1) GRASS GIS (Neteler, Bowman, Landa, & Metz, 2012; Neteler & Mitasova, 2008), (2) Quantum GIS (Hugentobler, 2008), (3) ILWIS/ILWIS Open (Hengl, Gruber, & Shrestha, 2003; Valenzuela, 1988), (4) uDig (Ramsey, 2006), (5) SAGA (Conrad, 2007; Olaya, 2004), (6) OpenJUMP (Steiniger & Michaud, 2009), (7) MapWindow GIS (Ames, Michaelis, & Dunsford, 2007), and (8) gvSIG (Anguix & Diaz, 2008).

In addition to these eight projects some more specialized, or localized, projects exist, e.g., the KOSMO project, which focuses on Spanish users, the German Kalypso project that focuses on modeling and management of water resources, and the Whitebox project, which provides tools for the analysis of raster data. There are also several other projects that are growing small user and developer communities, e.g., iGeoDesktop and OrbisGIS. For those projects it will be interesting to see how they are accepted by GIS users, what innovative concepts they can deliver, and how the software evolves over the next few years.

As Steiniger and Bocher (2009) point out, it is important to recognize that each of these projects concentrates its software development activities on a limited range of functionality. Hence, a certain set of GIS tasks may be accomplished easily with one FOSS4G application, whereas another FOSS4G solution may not

[2] Spring GIS (http://www.spring.org.br) could be member of that list, but unfortunately the INPE website and the referenced website have different license terms. One is not mentioning the right to change and distribute source code, while the other does allow it, using GPL 3.0.

provide the functionality due to its project goals (e.g., focus on functions for consistency and quality checks for validating geometries, printing maps, querying databases, etc.). This seems to result in the fact that many FOSS4G desktop GIS users switch between a number of desktop GIS to solve a range of tasks. In fact the recently founded gvSIG Community Edition project bundles gvSIG, GRASS and SAGA in one distribution.

5.2. Spatial DBMS

Spatial Data Base Management Systems (Spatial DBMSs) provide an alternative to storing geographical data in files. Usually a spatial DBMS is used if (i) large datasets, i.e., several thousands of geographical features such as buildings, roads or points of interest, need to be stored, queried, analyzed and updated, and (ii) if operations need to be performed in as short a time as possible. Modifying the definition of a spatial database given by Güting (1994), we can describe a Spatial Data Base Management System (Spatial DBMS) as (i) a DBMS, which (ii) offers spatial data types in its data model, i.e., geometry data types defined in the Open Geospatial Consortium (OGC) Simple Features Specification for SQL[3] (Herring, 2010), that (iii) offers a query language, and offers (iv) spatial analysis operations as defined in the OGC Simple Features Specification (Herring, 2011), such as intersection, union, etc. A spatial DBMS may further provide (v) spatial indexing structures, e.g., an R-Tree (see Samet, 2006).

In comparison to the wealth of desktop GIS projects there are only a few projects that work on FOS spatial DBMS. Usually the projects do not concentrate on developing completely new software but adding spatial capabilities to existing free DBMS. The dominating product, i.e., the free DBMS with the largest user group, is PostGIS (Obe & Hsu, 2011; Sherman, 2012), which provides spatial data types and analysis functions for the free PostgreSQL database. Besides the fact that PostGIS has the largest user base today and is being used in installations with high demands, for example at the French National Mapping Agency (IGN France) and Infoterra (a remote sensing image provider), PostGIS also provides the most feature complete implementation of the OGC Simple Features Specification among free DBMS software. Comparisons with proprietary Spatial DBMS, such as Oracle 11g Spatial, show that PostGIS is an alternative with respect to functionality, robustness, support and price (see for instance Greener, 2009; Obe & Hsu, 2010).

Competition for PostgreSQL/PostGIS comes from the MySQL Spatial project, which aims at adding spatial support to the free MySQL database, and the SpatiaLite project for SQLite. Currently MySQL Spatial provides a basic implementation of SFS SQL, but query and analysis operations used minimum bounding rectangles instead of the true geometries, hence those operations have low precision and may not be useful for everyone (BostonGIS, 2009; Ramsey, 2011). However, new spatial operations that use the geometry boundaries are under development and should come with MySQL Server 5.6 or 6.x. MySQL could be a strong competitor for PostGIS, as it has a large user base due to its position as standard DBMS for Internet applications.

SpatiaLite may be best described as FOSS alternative to the proprietary 'File Geodatabase' by ESRI. It realizes conformance to the OGC Simple Features Specification by utilizing the functionality of the GEOS geometry library (see Section 5.7). In addition to these three projects spatial support has also been developed for (i) Hibernate, an object relational persistence framework, (ii) for H2 DBMS by the different projects Hatbox, GeoServer and OrbisGIS, and (iii) spatial data support is underway for INGRES database systems. Of note is also the Jaspa project that adds Java-based spatial functionality to PostgreSQL (like PostGIS) and H2, including a topology validation system (Martinez-Llario & Gonzalez-Alcaide, 2011). Recent interest in non-SQL databases[4] such as CouchDB or MongoDB for storing spatial data has emerged as well. GeoCouch (Mische, 2011) is such a dedicated project for CouchDB, and also MongoDB offers methods for geospatial indexing.

5.3. Server GIS and WPS servers

For this article, our definition of server GIS does not refer to web severs for web mapping applications (see Section 5.9.1), but rather to server software that expose GIS functionality typically found in desktop GIS software. This functionality does not require direct interaction from a user via a user interface. For instance, the creation of new data or editing of geographic data requires direct interaction from a user, whereas analysis functions such as line buffering requires only the provision of a single parameter, which can be provided via user input, or estimated using an automated method. In this sense a Server GIS is software that (i) is used remotely, and (ii) provides access to functionality via web protocols such as the OGC Web Processing Service (Schut, 2007) and (iii) allows for the provision of data and input parameters from other web-based services (i.e., allows service chaining). Neun, Burghardt, and Weibel (2008) describe various data types that a Server GIS should be able to return, e.g., geometries, attributes (i.e., single values), single features, and complex data structures such as graphs.

The 52 North WPS is an OGC WPS conforming implementation of a server GIS developed by 52 North (Schaeffer & Foerster, 2008). Spatial analysis methods, i.e., processing functions, provided by the 52 North WPS can stem from the Sextante project (see Section 5.7), GRASS, the R project/software, and ESRI's ArcGIS Server. 52 North also distributes client plug-ins for the free desktop GIS' uDig and JUMP/OpenJUMP, and for OpenLayers, a web-mapping client library (see Section 5.9.2). A second implementation of the WPS standard is offered by "deegree" as part of their deegree3 web services architecture. deegree is a Java framework that can be used to implement a spatial data infrastructure. PyWPS also implements the OGC WPS standard (Cepicky & Becchi, 2007). PyWPS enables access to a wide range of analysis functions via GRASS GIS.

ZOO, founded in 2008, is as well a project that develops software that enables existing open source libraries to interact through the WPS services framework (Fenoy, Bozon, & Raghavan, in press). In addition to their ZOO Kernel, which manages and chains WPS, ZOO is developing an API that enables software developers to augment web GIS viewers such as OpenLayers (see Section 5.8.2) with advanced spatial processing capabilities. Similar to PyWPS and 52n WPS, ZOO provides access to GRASS GIS analysis functions. The grass-wps-bridge project is supporting the previously mentioned WPS servers and aims at delivering general libraries and programs to integrate GRASS per WPS. GeoServer, originally a web mapping server software, implements the WPS standard since Version 2.1. This will allow GeoServer users to access spatial analysis functions of spatial analysis libraries, such as Sextante and JTS (see Section 5.7).

Finally, as part of a university research project, a specialized type of server/web service was developed in 2005 to allow the use of cartographic generalization methods over the Internet. First versions of the service used the JUMP/OpenJUMP framework as a base for server and client implementation (Burghardt, Neun, & Weibel, 2005; Foerster et al., 2008). Later the project was made more generic and adapted to the then introduced WPS standard.

[3] SQL: Structured Query Language.

[4] To non-SQL databases is often referred as "NoSQL" databases, see: nosql-database.org.

5.4. Mobile GIS

We consider Mobile GIS to be GIS software that (i) runs on a mobile platform, e.g., a tablet PC, a Personal Digital Assistant (PDA), or a mobile phone, and that (ii) provides special functions that support data acquisition and redlining (updating) in the field. As such, the functions provided by a mobile GIS platform are generally a subset of those offered by most desktop GIS augmented with direct positioning capability via either an internal or external GPS. Costly analysis functions are rarely required when in the field, as these tasks can normally be performed later upon return to the office. Often specific, adaptable data entry functions are offered as well. They are useful since (i) data entry requirements are often predefined, (ii) users demand simple interfaces that allow data entry/editing to be performed in a timely manner (i.e., provide a large button for each type of data entry and limit the use of menus), (iii) work has to be accomplished under a range of environmental conditions – e.g., rain and sunlight can limit easy viewing of a mobile devices screen, and (iv) mobile devices offer also other input modes than mouse and keyboard, such as touch screen and pen. Special consideration with respect to mobile GIS may also be given to working with data off-line. That is, special functions and mechanisms for geographic data viewing, storage and transfer are provided.

To the authors knowledge there exists only two free general-purpose mobile GIS: gvSIG Mobile and Quantum GIS for Android. However, the latter project was started very recently in 2011 and seems to focus on Android tablets. There have been also experiments utilizing the desktop GIS GRASS on a PDA (Neteler & Raghavan, 2006) and OpenJUMP running on a tablet PC. Neither software provides additional functions to support GIS work in the field. Discussions among interested users and developers (on the OpenJUMP email list) have suggested that a mobile GIS tool can only be developed to a certain point when branching from an existing free desktop GIS platform. In particular the graphical display and user interfaces demand an alternative design perspective due to different hardware display features (small screen size, no or limited colours) and ergonomic conditions (e.g., no mouse, maybe a pen, sunlight, dirt, etc.). This, we suspect, may be the reason why few open source software projects exist to date. However, besides the two projects above, of additional note is Geopaparazzi for Android phones, which aims to support fast qualitative geologic surveys.

With the recent explosion of smart phone technology we anticipate that there will be a number of developments in the mobile domain. Montesinos and Carrasco (2010) analyzed the situation for mobile GIS from an open source perspective.[5] Besides "Mobile GIS" they also addressed "Mobile Viewers" with GPS data acquisition capabilities for smart phones – in particular gvSIG Mini (see also Martin-Reinhold, Huerta, & Granell, 2010) and tangoGPS/FoxtrotGPS.

5.5. Exploratory spatial data analysis software

So called Exploratory Spatial Data Analysis (ESDA) tools provide functions "[...] to describe and visualize spatial distributions; identify atypical locations or spatial outliers; discover patterns of spatial association, clusters, or hot spots; and suggest spatial regimes or other forms of spatial heterogeneity [...]" (Anselin, Cohen, Cook, Gorr, & Tita, 2000, p. 226). Anselin et al. (2000) further point out that such tools form a subset of general Exploratory Data Analysis tools with a special focus on geographical data. Data visualization methods that support such exploratory analysis are for instance parallel coordinate plots; scatter plots; box plots; and different thematic map creation methods (Anselin et al., 2006).

One of the well-known software packages for ESDA is GeoDA (Anselin et al., 2006). The original GeoDA software was available free-of-cost but was not free & open source. However, since 2011 version 1.0 of the FOSS—OpenGeoDa—is accessible and promoted. Other ESDA software, available from the GeoDA Centre, includes STARS (Rey & Janikas, 2006) and PySAL (Rey & Anselin, 2007). Several ESDA functions have been developed for the free R language and statistical computing environment. The GeoDa Centre webpage provides links to these projects and other software packages. Bivand (2006) and Bivand, Pebesma, and Gomez-Rubio (2008) provide a good introduction to spatial data analysis using R. A further project that aims to provide a software and development framework for geographical data visualization and analysis is GeoVista (Gahegan, Hardisty, Demsar, & Takatsuka, 2008; Takatsuka & Gahegan, 2002). Researchers at City University, London, worked on the HiDE software, an interactive tool for visually exploring categorical data using hierarchical layouts (Slingsby, Dykes, & Wood, 2009). It is interesting to note that all projects mentioned here have originated from research projects and are still related to research institutions.

5.6. Remote sensing software

Remote sensing software generally focuses on the modification and transformation (geometrical and spectral) of aerial and satellite image data as well as Lidar data. Such software can include photogrammetric tools (i.e., geometric) and remote sensing (thematic) analysis tools (Jensen, 2006) for image correction and filtering, geo-referencing and orthorectification, mosaiking, vectorization and image object extraction. Desktop GIS sometimes offer functionality that can also be found in remote sensing software. However, we see desktop GIS and remote sensing software as separate, in that desktop GIS focuses on the display, editing and analysis of attribute information attached to vector data and/or raster (image) data, whereas remote sensing software focuses on operations that alter, transform, and analyze images.

In comparison to the number of desktop GIS projects reviewed, we are only aware of a few FOSS remote-sensing projects. The most prominent example is probably OSSIM, which provides functions for image geo-referencing, orthorectification, mosaiking, etc. A second software developed mainly by Brazilian labs is InterImage (Costa et al., 2007; Pahl, 2003). InterImage provides a knowledge-based framework for image interpretation. Also founded and developed by Brazilian researchers is e-foto, an educational digital photogrammetric workstation (Mota et al., 2012). Unfortunately, the textbook covering the educational part of e-foto is currently only available in Portuguese. We hope that someone in the community will undertake the effort of translating the book to English.

The fourth project is Opticks, which is mainly developed by Ball Aerospace. Opticks aims to be a general remote sensing software with a focus on spectral image transformations and image analysis. The GNU Data Language (GDL) project does not provide a desktop like software interface as OSSIM or Opticks. But it does offer a free compiler for the Interactive Data Language (IDL) that is used for image processing. IDL forms the basis of the proprietary remote sensing software ENVI and allows image processing in an automated manner. The desktop GIS ILWIS and GRASS should also be mentioned here, since both provide several image processing and transformation functions, such as georeferencing and mosaiking.

5.7. Software libraries for GIS development

Software libraries provide functionality that can be used by other applications. That is they do not offer a graphical or

[5] See http://wiki.osgeo.org/wiki/GIS_Mobile_Comparison for more details.

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

command-line user interface that allows direct use of the functions. Examples may include methods that: perform cartographic projections; enable reading and writing of different data formats; or provide geographical analysis algorithms, etc. Libraries can only be used by coupling them to a host application, i.e., a desktop GIS or a GIS server.

It is hard to draw a line between libraries that are indeed GIS-related function libraries and libraries that may provide useful functionality in the wider sense of GIS use, such as functions for simulation and visualization. For instance several projects exist that develop software and libraries for (Multi-) Agent Systems (MASs) – which can be used to simulate spatial processes (see Steiniger & Hay, 2009) – but it is hard to classify them as GIS libraries. Therefore we present below a subjective listing of libraries that we know, consider to be noteworthy, and are of particular use for desktop GIS projects. Since libraries are often bound to a particular programming language (e.g., C++, Java, etc.) we have included this information.

5.7.1. Data input/output and conversion libraries

GDAL (raster) and OGR (vector) are two libraries that import and convert between different geographic data formats. The libraries are written in C/C++ and developed by the same project. Several free and proprietary desktop GIS use these two libraries, and their Python bindings play a significant role in current FOSS developments. Two further data abstraction and access libraries are FDO, developed by Autodesk, and GDMS by the ORBIS GIS team (Bocher, Leduc, Moreau, & Gonzales, 2008). The Java toolkit GeoTools provides a rich set of classes and functions for data import/export and manipulation. The SharpMap mapping library (C#) provides a set of adapters using the .Net framework for several geographical data formats and DBMS. However, we note that most of the tools are based on GDAL/OGR. The libLAS library (C/C++) specializes in the import/export of laser scanning/LIDAR data, using the ASPRS LAS file format (ASPRS – American Society of Photogrammetry, 2009). Lastly, the GeoKettle project (Badard, 2009) has extended the Java Pentaho Data Integration (Kettle) ETL (Extract, Transform and Load) software to enable geographic metadata to be read and then transform the metadata's associated data into various other formats. Also JEQL offers data load and transform functions, with syntax that is familiar to SQL users. As such, JEQL and GeoKettle provide similar functionality to Safe Software's proprietary Feature Manipulation Engine (FME).

5.7.2. Geometry libraries

Geometry libraries form the backbone of several desktop GIS as they provide the spatial data types and operations necessary for the analysis of geographic data. The most prominent library is the JTS Topology suite (in Java) and its C++ port GEOS. Both conform to the OGC Simple Features Specification (OGC SFS 2011). A further JTS port exists for the .NET framework, called NetTopologySuite. Besides the "JTS tribe" two other geometry libraries are noteworthy: CGAL is a C++ library developed by the computer graphics and computational geometry developer community. It provides a diverse range of algorithms for geometric analysis and the creation of various geometric data structures (triangulations, search trees, meshes, etc.). The Boost.Geometry library, formally the Generic Geometry Library (CGL), is a C++ library that provides a set of generic spatial operators designed for analysis of spatial objects. Analysis can be preformed using both two and three dimensional coordinates on planar or spherical coordinates systems, and between different geometry types. This means that one can use the same function to compute the distance between two points on a plane, or to compute the distance between a 3D point and a line on an elliptical datum such as WGS84.

5.7.3. Projection libraries

Proj.4 is a C/C++ library that allows a user to project geographic coordinates to a cartographic (planar) coordinate system and to transform data between different coordinate systems and datums. Proj.4 is used by many projects and therefore can be considered a FOSS4G core library. Recently the Proj.4 code has been ported to Java and was named Proj4J. GeoTools as well as deegree 3 also provide projection functions for Java applications, while Proj.Net provides some common map projection transformations for the .NET Framework. For those who wish to create and print maps using different projections, the Generic Mapping Tools (GMT) package may be of interest. The OSGeo MetaCRS project aims at coordinating the work among several of those projects.

5.7.4. Geographic data processing and analysis libraries

The Sextante project has successfully developed a Java-based framework for the analysis and processing of vector and raster data. Sextante can be accessed from within several desktop GIS (e.g., gvSIG, OpenJUMP and ArcGIS), is processing backend for WPS, and utilized by ETL software. The framework also includes graphical components that enable the creation of workflows similar to the ESRI's ModellBuilder. Two additional projects that provide libraries for image processing are ImageJ (Java) and ORFEO (C++). ORFEO has a strong focus on remote sensing applications and includes numerous functions for filtering, feature extraction, segmentation, classification and change detection. TerraLib (C++, Java and VB programming options) has been developed to support geographical analysis and forms the basis of the Brazilian National Institute for Space Research (INPE) developed TerraView open source GIS (Câmara et al., 2008). TerraLib can connect to various spatial DBMS and is OGC compliant. The software provides a number of spatio-temporal data types and operations, provides functions for Cellular Automata (CA) modeling, and can be coupled with R for statistical analysis.

While not strictly an analysis library, PAL (Ertz, Laurent, Rappo, Sea-Tang, & Taillard, 2009) is a useful cartographic tool for label placement. It provides several algorithms for the placement of names (e.g., road names or house numbers) on a map that try to avoid overlaps among labels. Lastly, we refer readers to work by Simonetto and Follin (2012) that lists several packages for the analysis of Synthetic Aperture Radar (SAR) images.

5.7.5. Other libraries

Here we list a few libraries that may be of use for customized development that typically occurs within a research environment: (i) the math libraries Java Matrix Package (JAMA), JMathTools (Java) and the GNU Scientific Library (C++), (ii) the graph libraries JGraphT (Java) and Boost (C++), and (iii) the Java chart library JFreeChart.[6]

5.7.6. General frameworks

There are several general frameworks that provide a number of functions and classes for development of new applications. We have mentioned some of them already above, but want to highlight them here again. For Java development programmers should consider GeoTools (Turton, 2008), Puzzle GIS, and OpenMap. For .NET developers SharpMap may be of interest. In 2010 the developers of MapWindow GIS separated their project into MapWindow, as an application, and DotSpatial, as the underlying development framework (C++/.NET development). Hence, there are now two frameworks for .NET. Python developers and web/JavaScript developers should have a look at GeoScript. We finally note that

[6] Note, using Internet search engines (e.g. Google.com or Bing.com) with the project/library names mentioned should help to find related projects.

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

numerous desktop GIS are quite flexible for developing new applications, e.g., Quantum GIS for C/C++ and Python development, and uDig and OpenJUMP for Java development.

5.8. GIS extensions, plug-ins and APIs

GIS *extensions* add functionality directly to a particular software. For instance in the proprietary world, ESRIs Spatial Analyst Extension adds functions for (raster) analysis to ESRIs ArcGIS Desktop product. *Plug-ins* act in the same manner, but they typically add a narrow set of abilities, e.g. one particular raster analysis method, whereas an extension can add a wider set of new abilities, e.g. a toolbox of raster analysis methods. For that reason in OpenJUMP an extension is actually considered a collection of plug-ins. However, we could not find a reference in the literature that sufficiently explains the difference between plug-in and extension.

The term plug-in also describes the mechanism used to add functionality. That is, a plug-in is a separate program that cannot be executed on its own and instead utilizes software interfaces of the host program/software for control and data access. In contrast to the use of a plug-in mechanism/interface, the functions provided by a library usually need to be accessed via source code and do not call functions of the main application.

Plug-ins can be seen an alternative option to the use of scripts, that also add custom functionality to an application. But scripts have to be executed every time the software is started. When generating scripts or new plug-ins developers often utilize an 'Application Programming Interface', short: *API*. APIs offer methods that are frequently used in customized programming, such as data retrieval and display methods. This avoids that frequently used processes are re-programmed every time and reduces complexity by hiding functions that a plug-in or script programmer does not need to know about. Also software may offer an API to allow customization with different script/programming languages.

Almost all desktop GIS, such as Quantum GIS, uDig, OpenJUMP, gvSIG, etc. offer plug-in mechanisms and provide listings of available plug-ins. As there is a wide variety of extensions and plug-ins for several free desktop GIS we will not list those. Instead we refer the interested reader to the web pages of the desktop GIS. APIs that allow customization of desktop GIS and web map servers with the (script) language Python have become widespread in the last few years. A Python API, or its Java complement Jython, is available in Quantum GIS, GRASS, MapServer, gvSIG, OpenJUMP, etc. However, a particular Python program, in form of a script or plugin, works only with the software it was programmed for and not across different software.

5.9. Software for internet mapping applications

Under this category we summarize software that enables a developer to deliver (via a server software) and view (via a client software) geographic data and maps using standard internet protocols. In particular we address software that supports the Open Geospatial Consortiums (OGCs) Web Mapping Service (WMS) and Web Feature Service (WFS) standards. Earlier, several 'thick' web client projects existed that worked on separate software with that the user can view and edit geo-data over the internet. Examples for such thick clients are the free software NASA WorldWind and the proprietary Google Earth. But in the last 5 years new technologies emerged that allow to offer the same functionality in normal web browsers (what is called "thin client"). For instance, one project that aims at bringing navigate-able 3D maps, which were previously only known from NASA WorldWind, into web browsers, is the OpenWebGlobe SDK. Hence, we decided that we will not address the thick web client category, but focus instead on free software suites that aid the development of web mapping applications that can be run in a normal browser.

5.9.1. Web map server

In essence a web map server is a specialization of a standard web server (Milosavljević, Dordevic-Kjan, & Stoimenov, 2008) such that it can provide Web Mapping Services (WMS; ISO, 19128, 2005; de la Beaujardiere, 2004), Web Feature Services (WFS; ISO/DIS 19142, 2009; Vretanos, 2005a), or Web Coverage Services (WCS; Baumann et al., 2008). Each of these mapping services provides different views of spatial data. A WMS returns a geo-referenced rasterized map (layer). In this context a map is considered a two-dimensional visualization (according to a predefined style) of features and can be delivered using common formats such as jpeg or tiff. The service does not return the actual geographic features. By default a WMS serves one or more styles per layer. Defining a style on the client side and sending this style as part of the WMS request to the server will result in a map with a user-defined style. The styles used for rendering the map need to be specified in a Style Layer Descriptor (SLD) document (Lupp, 2007).

The WFS can be used for selecting, inserting, updating and deleting features – and enables geographic and attribute filtering of geospatial features. It differs from a WMS in that it provides access to discrete objects, i.e. vector features – and not only a complete "map". It additionally offers INSERT, UPDATE, DELETE, etc., capabilities for the vector features. The OGC filter encoding specification (Vretanos, 2005b) is required for filter definition to select only a subset of geographic objects (i.e., features) in a database. It describes the XML to be used to define spatial and attribute filters. The selected geo-features are returned in GML (Cox, Daisey, Lake, Portele, & Whiteside, 2005) format.

The WCS is similar to a WFS in that it provides direct access to geographic features, but unlike the WFS that returns discrete geospatial features (i.e., objects such as a building or a street light) the WCS can return either whole coverages, a set of features, or a grid coverage. A grid coverage can be an aerial photograph or elevation data, that typically represent space-varying phenomena.

As such, a web map server essentially offers a mapping service that provides access to spatial data. Server GIS, on the contrary, focus on the provision of analytical methods for the examination and exploration of spatial data. Should a web map server provide a WPS interface (see Section 5.3), then a web map server and a Server GIS can be coupled together via the WPS.

The best-known web map servers are (UMN) MapServer (Kropla, 2005) and GeoServer (Erle, Gibson, & Walsh, 2005). Both solutions are comparable to similar proprietary solutions with respect to functionality – and both packages offer comparable performance (Aime & McKenna, 2009) and tile–caching capability[7]. These servers offer vector and raster support, and conform to a number of OGC web mapping standards including WMS, WFS, WCS, GML, and SLD. MapServer runs as a Common Gateway Interface (CGI) application within the Apache web server environment, whereas GeoServer is built on Java technology and runs in an integrated Jetty or Apache Tomcat web server environment. A recent review by Ballatore, Tahir, McArdle, and Bertolotto (2011) suggests that MapServer is preferred in terms of performance, scalability, and stability, whereas GeoServer is easier to extend, and follows the various web mapping standards more closely. MapGuide Open Source offers a web map server component as part of their web-mapping suite. MapGuide OS has evolved out of Autodesk's proprietary MapGuide 6.5 software, conforms to OGCs WMS and WFS specification, offers

[7] Tile caching allows the fast delivery of pre-rendered map sections to WMS web mapping clients using predefined map scales and image (tile) sizes. Besides built-in tile cache solutions a well-known open source tile cache software is TileCache (tilecache.org).

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

a broad range of authentication and administration tools, and its own built-in tile-cache. lat/lon GmbH offer a web map server as part of their deegree application framework. deegree has a strong European user base and has been the reference implementation for several OGC standards (Lupp, 2008). Recently, there have been a number of projects added to this arena such as TinyOWS and QGIS Server. For further details we refer to OSGeo's Benchmark 2010 and 2011 wiki (wiki.osgeo.org/wiki/Benchmarking_2011).

Finally, we deem it important to mention two further server projects that provide a REST-based approach to data insert, update and delivery. RESTful approaches treat data as a resource, and representations of resources can be uniquely addressed by a universal resource identifier (URI). The earlier mentioned GeoServer, as well as the FeatureServer and GeoRest implement this type of data access. However, all three systems also offer WFS capabilities and provide data in various contemporary web formats such as GeoJSON, KML and GeoRSS.

5.9.2. Web map application development frameworks

Technological advancements, such as browsers that support scripting languages natively, and standards, such as Cascading Style Sheets (CSS), Asynchronous JavaScript And XML (AJAX), and HTML 5, nowadays allow that applications developed for web browsers can offer the same functionality as desktop software. The software and application development frameworks listed below enable the development of web mapping applications that run in normal web browsers.

OpenLayers implements a JavaScript API for visualization of spatial data in a web browser without the need for server-side components. It implements industry standard methods, WMS, WFS, etc., for accessing geospatial data using components from Prototype. Protoype is a AJAX library that introduces class-based design within a JavaScript development environment (Orchard, Pehlivanain, Koon, & Jones, 2009). OpenLayers offers spatial visualization and manipulation tools to simplify the development of rich web-based geographic applications (Hazzard, 2011). The ReadyMap suite adds 3D visualization capabilities to OpenLayers. But it includes other tools for publishing web maps as well. A focus on navigate-able 3D maps has also the OpenWebGlobe SDK project by Swiss researchers, which we mentioned as possible successor for thick clients such as NASAs WorldWind.

MapFish is an open source web mapping development framework that extends the Pylons development framework and provides support for spatial data. MapFish aggregates the power of OpenLayers and GeoExt to allow developers to create advanced web mapping applications with print, search and edit capabilities. OpenLayers and GeoExt ensure that MapFish is OGC compliant with respect to WMS, WFS, WMC, KML and GML standards. GeoExt extends user interface widgets offered through ExtJS to understand spatial features. In addition to the primary Python/Pylon implementation framework, MapFish can be also deployed within a Ruby/Rails or PHP/Symfony framework.

There are numerous other web map application development environments that all offer similar capabilities to OpenLayers and MapFish: Leaflet is a JavaScript library for the creation of interactive maps by the founders of OpenStreetMap. Interesting for developers is probably the focus on desktop and mobile web browsers, and its use of HTML5. The open source mapping framework OpenScales focuses on a complete different basis technology, since it is written in ActionScript 3 (an object oriented language) and Flex. Flex is Adobe's cross-platform development environment for Adobe's Flash multimedia platform for desktop and mobile (Android, iOS) applications. Also the ModestMaps library was further developed in the past years and forms together with the Wax extension a competitive alternative to the above mentioned Leaflet and OpenLayers libraries. However, ModestMaps/Wax and Leaflet currently compete with OpenLayers only with respect to the display of map tiles, because OpenLayers offers much more functionality when it comes to interactive and vector-based mapping tools.

Leaving the software group of client-side mapping libraries, several other web-mapping development frameworks should be mentioned: GeoMOOSE and i3Geo are free JavaScript frameworks for displaying geographic information. Both build upon other open source projects, particularly OpenLayers and MapServer. Mapbender is a server and client framework for spatial data infrastructures. The software is implemented in PHP, JavaScript, and XML and is licensed under GNU GPL and Simplified BSD licenses. GeoMajas provides a platform for server-side integration of geospatial data. For that reason, good editing capabilities for geometry and attribute data as well as data management functions are a focus of the project besides data visualization. Supported OGC standards are WMS and WFS, and database connectors are provided by GeoTools or Hybernate Spatial. MapStraction is an open source JavaScript mapping library that allows developers to create applications independently of a particular mapping framework, i.e., developers can easily switch in and out mapping frameworks as their needs demand. MapStraction supports many of the major mapping frameworks (Google, OpenStreetMap, Yahoo, Bing, etc.). See Duvander (2010) for more information. Already listed earlier SharpMap is mentioned here again as it can help ASP.NET developers in creating online mapping applications.

To assist in the development of webmap applications – toolkits such as the Google Web Toolkit (GWT) can enhance productivity. GWT is a development toolkit for building and optimizing browser-based applications. The toolkit can assist with cross-browser development issues, XML HTTP requests and JavaScript implementation by removing some of the complexities of these technologies. GWT is licensed under the Apache License v.2.0, but also includes a number of third party tools licensed under a range of open source licenses. Lastly, tile-renderers may be needed as well to create the maps to be served, as an alternative to WMS services. We know of two software projects with a growing user base: Mapnik and TileMill. Mapnik is a renderer particular created for generating maps from OpenStreetMap data. TileMill is related to the MapBox service and does not focus on OpenStreetMap data only.

In the 3 years that we wrote, re-wrote and revised this manuscript the section on web mapping has been the one with the most activities and additions – besides the WPS section. New projects like Leaflet, Mapbox and ReadyMap, that have been all started by companies, suddenly appeared on the 'map' and gained strong interest when license terms of the Google API were changed in 2011 (see below). Hence, with the technological advancements we expect much competition and fluctuation in that area for the years to come.

5.9.3. A note on web-mapping APIs such as Google Maps

In presentations people have asked us why our FOSS4G overview does not include popular web mapping Application Programming Interfaces (API's), such as Google Maps, Yahoo! Maps or Bing Maps, or even the Google Earth application—even though they are 'free'. The reason is, that these map API's are only free-of-cost in certain situations, and that their licence agreement imposes restrictions on users that limit the APIs uses to certain types of applications. For example these APIs are not free for commercial uses, and private users are restricted in the daily frequency of use (number of map requests) of the services offered through these API. Payment of licensing fees for commercial applications generally provides the commercial entity with a web mapping platform free of third party advertisements, as this is generally how the API providers generate income. A recent example for a license change is the Google Maps API. Until October 2011 there have been no

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

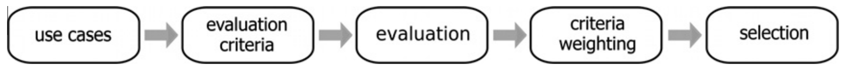

Fig. 2. Possible software selection process.

transaction limits for the maps generated with the API, whereas now there are (Google Geo Developers Blog, 2011).

Additionally we note that these API's are *not free* in terms of access to algorithms and procedures called via the various public API functions – i.e., the APIs are not open source APIs. Thus, these web-mapping tools work as black boxes and do not give users the freedom to study and modify them. Furthermore, control over these APIs is maintained exclusively by the API providers as their licenses enable them to terminate the services at their discretion, although given their business objectives we believe that this is unlikely.

We strongly recommend that all potential web mapping API users (i) read the license terms provided with all APIs, (ii) note the accessibility of the API code base, i.e., is it free in terms of free and open source software? Further, (iii) potential users should note all limitations on the use and distribution of maps generated by the API. This is important since typically, background data/imagery is provided by third parties and they often impose limitations on the use of the data/maps by the API provider (Google Maps Legal Notices, 2012; Microsoft, 2012). Those limitations are transferred to all API users. Also, (iv) licenses should be studied with respect to storage and use of user provided data that is overlaid on API provided data. This may be critical because the API license may give the API provider the right to use the user provided data in any way they see fit (see Google API Terms, 2012, sec. 11). Some API licenses have even claimed copyright to derived works – a mashup combining data from various sources (see clause 2.1 in British Ordnance Survey, 2010). This raises at least two questions, (a) given the data license associated with your data, are you able to offer exclusive use licenses to others? (b) Do you want to give a third party the right to reproduce, alter, transform, etc., your data? For additional discussion on these issues we refer to blog and forum discussions by Dickson (2008) and Parsons (2008) on the use of mapping data of the British Ordnance Survey within Google applications.

6. Selecting free and open source GIS software

There are well-known benefits of open source software, such as cost savings, vendor independence, and open standards. But there are also a number of barriers, such as knowledge barriers, legacy integration, forking, sunk costs and technology immaturity (Nagy, Yassin, & Bhattacherjee, 2010). Knowledge barriers often exist because potential users are not aware of the availability or relevance of FOSS4G, nor of the technical requirements necessary to make effective use of FOSS4G. Legacy integration has been problematic when using FOSS4G, but in recent years there have been numerous middleware solutions developed that have minimized this barrier. Forking of a project occurs when different groups develop software, and the resulting components no longer interoperate with each other because the original project failed to impose adequate standards. This can result in the demise of smaller open source projects that fail to maintain a large enough developer base. This particular barrier can be addressed through the development of self-managed standard groups within a project. Sunk costs arise due to prior investment in proprietary software. The simplest way to address this barrier is to compare future cost streams of maintaining proprietary software versus open source software. With regard to technical maturity, it is imperative that software be evaluated independently against case studies derived from an organization's core functionality.

As an organization it would be folly to adopt FOSS4G just because other organizations do, or based solely on claims in the literature. Instead, the decision to adopt FOSS4G should be based upon careful considered rationale (Ven, Verelst, & Mannaert, 2008). The selection process, be it for business, research or teaching, should be based on a rigorous software evaluation process. Among the criteria that should be applied during the evaluation process are for instance: software functionality, software stability, platform support, market share, credibility (and branding) of the software manufacturer, as well as the size of the support and distribution network. While these criteria are common to the selection of proprietary software – with respect to free and open source software other criteria are also important. In particular, consideration should also be given to the software project that created the FOSS4G software, since the status of the project will influence software support, software evolution (in terms of functionality and domain of use), and longevity (see also Cruz, Wieland, & Ziegler, 2006).

A formal software selection process should consist of five steps (Fig. 2):

(i) Develop software use cases for own context (or "user stories"[8]).
(ii) Establish a set of evaluation criteria based on the use cases.
(iii) Perform the software evaluation with respect to the established criteria.
(iv) Develop a weighting criteria according to application context (note, weighting is intended to be flexible so as to allow for different contexts).
(v) Select the software based on results of the evaluation and weighting scheme.

This proposed five step process was derived from the method used by the company Atos Origin (2006), covering the steps (1) definition, (2) evaluation, (3) qualification and (4) selection, as well as the four-step process outlined by Sveen (2008), consisting of (1) define usage scenarios and requirements, (2) gather candidate projects, (3) create evaluation sheet, (4) rank projects and select.

Sets of evaluation criteria for free (GIS) software have been proposed by several authors, e.g., Reid and Martin (2001), Ramsey (2007), Wagner (2006), Cruz et al. (2006), Sveen (2008), Steiniger and Bocher (2009). Out of those publications we have compiled a list of criteria, presented in Table 2. In the table we have sorted the criteria according to their relevance if the software is to be used for business, research or teaching.

Of particular relevance to FOSS4G are the Organizational and Legal criteria in Table 2. Two important aspects of the Organizational category are "Size/existence of user and developer community" and "adoption by companies and administration". Both criteria will influence (i) how good support from the community will be, and (ii) if support offered by companies is available. In simple terms, the more users and developers a software project has, the higher the chance that user questions to the community will be answered in a timely manner, and the higher the chance that users experiencing similar issues will help each other.[9] If users

[8] "User stories" are used in agile software development and have the structure "As a ⟨role⟩, I want ⟨goal/desire⟩ so that ⟨benefit⟩" (Ambler, 2009).

[9] We like to note here, that the website "stackexchange.com" is a useful general help website where users can ask questions to other users. It also has a GIS section: gis.stackexchange.com.

Table 2
Project and software evaluation criteria. For details see text.

	Business	Research	Teaching
GIS use objectives	Accomplish certain routine tasks in an efficient way	Accomplish necessary tasks (e.g., related to field work) and develop new concepts and algorithms	Explain and demonstrate concepts and methods
Evaluation criteria			
Functional	• Required functionality • 'Nice to have' functionality • Project road map/focus • User level (viewer vs. editor vs. analyst) • Usability • Customization options	• Required functionality • 'Nice to have' functionality • Customization options	• Required functionality • Usability • Customization options
Technical	• Supported Operating Systems • Programming language • Reliability (maturity, robustness) • Maintainability (ease of modification, clear structure)	• Supported Operating Systems • Programming language • Maintainability (ease of modification, clear structure) • Precision • Reproducibility	• Supported Operating Systems • Reliability (maturity, robustness) • Maintainability (ease of modification, clear structure)
Organizational	• Size/existence of user and developer community • Project drivers and lead • Adoption by companies and administration	• Size/existence of user and developer community • Project drivers and lead	• Size/existence of user and developer community • Adoption by companies and administration
Support	• Company/hotline support • Documentation • Email lists • Forums • Wiki	• Documentation • Email lists • Forums • Wiki	• Documentation • Email lists • Forums • Wiki • Workshops and user conferences
Economical	• Migration costs • Training costs • Maintenance costs	• Training costs • Maintenance costs	• Maintenance costs
Legal	• License type (GPL type vs. LGPL type vs. BSD)	• License type (GPL type vs. LGPL type vs. BSD)	

Table 3
Questions to the future software user that can help to select software for daily GIS use (i.e., not software development).

Guiding questions
Functionality
• Can you deliver detailed case studies/uses cases that show what the user should be able to do with the software and what they should not be able to do? (*Note:* what the user should *not* be able to do adds requirements.)
• For desktop GIS or web-based service infrastructures: Do you want to just create maps, or do you need analysis functionality as well? I.e., do you need a simple data viewer, or do you need a data editor and analysis functions too? See for instance the GIS tasks listed in Table 1
• Are your data in raster or in vector format?
• Are your data stored in a database or files? Do you want to write your data to a database or files?
Platform
• Will your users work with Microsoft Windows, MacOSX, or Linux only? Do you prefer a cross platform product that runs on MacOSX and Linux as well? (*Note:* this may become important when adding custom functionality)
Support
• What programming languages are your users and developers comfortable with? (*Note:* this is important for user support on email lists and forums)
• Is hotline and emergency support needed?
• Do your functional requirements need custom developments? (Notes: If others are contracted for customized development, then these external developers need to be available. Also, if community developers or contractors should help, then the software should support easy implementation, e.g. by offering a plug-in mechanism and an API.)
Other
• Is using free-of-cost but proprietary GIS an option, or even low-cost but proprietary solutions? (*Note:* if yes, the set of software to be considered may be much larger.)

are coming from companies and administration (and not only from research or education) then there is also a high probability that professional paid support, in terms of hotline support, and training is available.

The organizational criterion "project drivers & lead" is important, since a project that is broadly adopted or has strong leadership will be more likely to continue to be developed and ensure regular official software releases. If for instance a software project is driven/led by a company or a public administration, then the ability of the user community to influence software evolution may be lower than that for volunteer projects. However, a company lead can have also advantages. For instance testing will probably be more rigorous and consequently the stability of software releases may improve. Also, user documentation is more often available together with the software release and long term project plans have been developed.

Another criterion of importance to FOS GIS software evaluation is the legal criterion: "License type (GPL type vs. LGPL type vs. BSD)". The license under which the free software is distributed becomes important if custom developments are carried out, in terms of changes and extension of the original software, and if those developments are to be distributed to an organizations' own customers. If these developments should be kept private, then GPL-like and LGPL-like licenses cannot be used, since they require that the (non-in-house) source code for all distributed software be published. Similarly, license aspects are important when free software is coupled with proprietary software. For instance, it is not possible to distribute a proprietary GIS (e.g., ESRI's ArcGIS) that utilizes functions from a library that is covered by the GPL license. Because the GPL license terms would require that the source code for ArcGIS also be published. However, if the LGPL licence or the X/MIT license is used, then such bundling of proprietary and open source software is possible. ArcGIS, for instance, is using the GDAL library that is distributed under the X/MIT style license. We refer the interested reader to Gangadharan, D'Andrea, De Paoli, and Weiss (2009) for a discussion on licence compliance issues on the example of GRASS GIS.

In addition to the criteria outlined in Table 2, we believe it is beneficial to start a software evaluation with a basic set of questions that are linked to these criteria. This should help to narrow the set of candidate software, which can then be evaluated in more detail. These basic questions are listed in Table 3. For readers interested in a more detailed discussion on this topic we refer to Sveen (2008), who developed and applied an evaluation methodology for desktop GIS in a company setting.

The traditional software evaluation process, as we have outlined it above, will usually strive to obtain *one* "winning" software. However, we note that several users of FOSS4G applications whom we know personally employ a "multi-GIS use strategy". That is they use a particular GIS for a specific task, of which the user thinks it is best for accomplishing the task. As all tools are free and no additional licence costs emerge (besides training and maintenance costs), this approach does enable users to develop efficient work flows that produce the best possible results, and overcome the limitations of adopting one particular software. Furthermore, this multi-GIS use strategy will also further the literacy of the GIS user since knowledge of "GIS concepts" and not of "software buttons" is requested. However, with the multi-GIS strategy the user needs to become familiar with several tools, and needs to transfer the data between the tools.

7. Further resources on FOSS4G

In this section we aim to review the recent literature on FOSS4G. Whereas it is recommendable to read the material we referenced for each particular software – there is also a growing body of literature that introduces FOSS4G, compares capabilities of software, or demonstrates the building of more complex applications utilizing several open source software. A beginners guide to GIS and FOSS4G is for instance the book by Sherman (2012), now in its second edition. The book by Hall and Leahy (2008) gives a good account on the projects and topics about 5 years ago. Journal articles that introduce FOSS4G to a research audience have been written by Jolma et al. (2008a, 2008b), Rey (2009), Steiniger and Hay (2009), and Steiniger and Bocher (2009). In particular Jolma et al. (2008a) looked at the usefulness of FOSS4G for ecological applications, and Rey (2009) analyzed the value of the free and open source idea for the spatial analysis community. In Steiniger and Hay (2009) free and open source GIS is introduced to landscape ecologists and available tools are surveyed.

Steiniger and Bocher (2009) analyzed the state of FOS desktop GIS developments in 2007/2008. The material is by now 5 years old, and consequently some desktop GIS advanced a lot in their functionality (and market presence) during that time. However, their general discussion on software licences and their discussion on the value of FOSS-based research and teaching for GIScience are still contemporary. The importance of free and open source principles for science has recently received attention in journals such as Science and Nature. Among the articles we recommend to fellow scientists are Ince, Hatton, and Graham-Cumming (2012), Morin et al. (2012), Rocchini and Neteler (2012), and Barnes (2010). The first three publications make a strong point against research creating proprietary – 'black box' – programs that hinder scientific advancement and testing. Barnes (2010) argues that also 'unready'/'unpolished' software and programs produced in research should be published, as it allows other researchers to learn from and to build on; even from unpolished programs.

There are also articles that discuss and compare FOSS4G software for particular applications. For instance, Chen, Shams, Carmona-Moreno, and Leone (2010) assess free GIS software for water resource management, and Pieper Espada (2008) compares software for use in land administration systems. A broader view take Sanz-Salinas and Montesinos-Lajara (2009) in their general review of major GIS software projects, and Steiniger and Hunter (2012) that look at software to implement Spatial Data Infrastructures (SDIs). Ballatore et al. (2011) analyze free software for building a system for the delivery of location based web mapping services. Finally, Donnelly (2010) compared several free desktop GIS with the proprietary ArcGIS software by analyzing their capabilities to create thematic maps.

In addition to the articles and books we wish to note several web sites that collate information on existing and developing FOSS projects. These include: freegis.org and opensourcegis.org who list a variety of GIS software along with a short description and links to project web sites. The Open Source Geospatial Foundation (osgeo.org) sponsors and supports the use of FOSS4G software, and provides a platform for cooperative software development. Readers with an interest in spatial databases should also visit bostongis.com for articles that introduce and compare FOSS4G databases with similar proprietary products.

8. Future developments

From our perspective the use of Free and Open Source Software for GIS (FOSS4G) by GIS users is transitioning from the 'early adopters' stage to the 'early majority' stage (see Rogers, 1995). This is visible in an increase of software download rates, an increase of participants at local FOSS4G conferences (e.g., FOSSGIS Germany), and FOSS4G consulting companies with full order books – even during the recent economic crises. Additionally, new business strategies around open source GIS software have been introduced focusing on services for GIS/Mapping infrastructures over the Internet or an Intranet. What facilitates the transition from the 'early adopters' stage to the "early majority" stage is the fact that certain FOSS4G software products can compete with proprietary software products. This has been shown by several software evaluations, e.g.: Steiniger and Hay (2009) and Donnelly (2010) for desktop GIS, Greener (2009) for spatial DBMS, Tweedie (2009) for web map/image server, and Delmont (2012) for online mapping services. Hence, a lack of software maturity is no longer a hindrance to growth of the user community. However, Câmara, Vinhas, Modesto, and de Souza (2012) have pointed out intrinsic challenges to the FOSS4G development community, such as leadership, code stability and a lack of shared concepts for analysis and processing – that may hinder growth.

But what are future developments likely to be? We see three characteristic features for future FOSS4G development, besides a growth in users:

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003

(i) Emergence of new projects (see our review above).
(ii) Consolidation, due to projects that stop their developments (e.g. MapBender and MonoGIS).
(iii) Collaboration among projects.

With respect to the emergence of new projects we doubt that there will be many new successful projects that are founded by a group of volunteers with similar interests (such as QGIS). Because making a volunteer project successful requires (i) skilled project leaders and (ii) a passionate group of people that follows its objectives over several years – these are two conditions that are difficult meet. It is more likely that new projects will emerge either from a business background, i.e., the projects are created by companies and probably stimulated by public funding through public administrations (e.g., gvSIG, JUMP/OpenJUMP), or from a university background, i.e. university projects that develop innovative concepts. It is probably noteworthy that existing software, which was/is free-of-cost, has been transformed into free and open source software projects. An example here is the desktop GIS ILWIS, and the work being undertaken by INPE Brazil to make the free-of-cost desktop GIS Spring open source.

The consolidation of projects will likely occur in situations where one projects ceases further development, either due to (i) competing solutions, (ii) the project filled a niche that was too small to sustain itself, or (iii) poor leadership, and is then absorbed into another complementary project. The difficulty with consolidation is that some users will need to look for alternatives. But, because the source code is open, anybody is able to continue work on it. Furthermore, people that leave the project may contribute their knowledge to other projects, which will strengthen those projects and make them more competitive (against proprietary products). This has been the case with the OpenLayers project that has been joined by team members from MapBuilder, Chameleon, and ka-Map (Adams, 2009).

There is an increasing level of collaboration among projects today. This is in part because several projects use the same geometry libraries (JTS, GEOS, NTS) or interoperability libraries (GDAL/OGR). Hence, a continuation of the development of these libraries is a common interest. Similarly, with the introduction of new standards, such as the OGC Styled Layer Descriptor standard (OGC SLD), the projects are striving to make their implementations compatible with each other – which is eased by the availability of open source code. Such collaboration may also lead to innovation and new standards. Perhaps, the GeoRSS standard can be considered as an example for a community driven standard. Frequently, new collaborations are established during workshops and so-called "code sprints". From our point of view the exchange between the projects has greatly improved with the establishment of the OSGeo foundation (osgeo.org). To date the OSGeo serves as a common point of contact, and offers infrastructure tools to member projects. The past has shown that these tools have facilitated project collaboration and development.

We hope that our project review helps others to adopt FOSS4G for use, and even encourages users to participate in FOSS4G projects. The major benefit for business is the lack of a license fee – though maintenance and training costs will continue to exist. The major benefits for research and teaching are the price tag, the possibility to peer-review the source code/implementation, and the facilitation of an exchange of new knowledge on algorithms and methods. Hence, for these domains we believe it is logical that FOSS4G be considered, examined, and tested against proprietary GIS software when you next consider extending the GIS software needs. A good opportunity for testing gives the OSGeo LiveDVD project that distributes several free GIS software on a bootable Ubuntu DVD (live.osgeo.org). Finally, for those that plan developing FOSS or participating in a project the book "Open Advice – FOSS: What we wish we had known when we started" by Pintscher (2012) should be of interest.

Acknowledgments

During the 3 years of writing, re-writing, re-searching, and editing the manuscript the authors received support and funding by the Swiss National Science Foundation (SNF), GEOIDE and the Neptis Foundation. We also thank those that encouraged us year by year to write (and finish) this overview, and the two reviewers that gave helpful comments on additional projects and suggestions that improved the manuscript.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003.

References

Adams, T. (2009). There is no alternative to OpenLayers...? In *Presentation at FOSS4G 2009 conference*, Sydney, Australia. <http://maps.terrestris.de> (Accessed 17.01.12).

Aime, A., & McKenna, J. (2009). Web mapping performance shoot-out. In *Paper presented at FOSS4G 2009 conference*, Sydney, Australia. <http://slideshare.net/gatewaygeomatics.com/wms-performance-shootout>, see also: <wiki.osgeo.org/wiki/Benchmarking_2009> (Accessed 17.01.12).

Ambler, S. W. (2009). *Introduction to user stories*. <agilemodeling.com/artifacts/userStory.htm> (Accessed 17.01.12).

Ames, D. P., Michaelis, C., & Dunsford, T. (2007). Introducing the MapWindow GIS project. *OSGeo Journal, 2*. osgeo.org/journal.

Anguix, A., & Diaz, L. (2008). GvSIG: A GIS desktop solution for an open SDI. *Journal of Geography and Regional Planing, 1*(3), 41–48.

Anselin, L., Cohen, J., Cook, D., Gorr, W., & Tita, G. (2000). Spatial analyses of crime. *Criminal Justice, 4*, 213–262.

Anselin, L., Syabri, I., & Kho, Y. (2006). GeoDa: An introduction to spatial data analysis. *Geographical Analysis, 38*, 5–22.

ASPRS – American Society of Photogrammetry and Remote Sensing (2009). *LAS specification, v. 1.3 – R10*. Bethesda, MD: The American Society for Photogrammetry & Remote Sensing.

Atos Origin (2006). *Method for qualification and selection of open source software (QSOS) version 1.6*. <www.qsos.org/?page_id=3> (Accessed 17.01.12).

Badard, T. (2009). GeoKettle: A powerful open source spatial ETL tool. In *Paper presented at the FOSS4G 2009 conference*, Sydney, Australia.

Ballatore, A., Tahir, A., McArdle, G., & Bertolotto, M. (2011). A comparison of open source geospatial technologies for web mapping. *International Journal of Web Engineering and Technology, 6*(4), 354–374.

Barnes, N. (2010). Publish your computer code: It is good enough. *Nature News, 467*(7317), 753.

Baumann, P., Cox, S., Donéa, L., Evans, J., Herring, J., Keens, S., et al. (2008). *Web coverage service (WCS) implementation standard*. Open Geospatial Consortium.

Bivand, R. S. (2006). Implementing spatial data analysis software tools in R. *Geographical Analysis, 38*(1), 23–41.

Bivand, R. S., Pebesma, E. J., & Gomez-Rubio, V. (2008). *Applied spatial data analysis in R*. New York, NY: Springer.

Blasby, D., Davis, M., Kim, D., & Ramsey, P. (2002). *GIS conflation using open source tools*. The Jump Project. <http://citeseerx.ist.psu.edu>.

Bocher, E., Leduc, T., Moreau, G., & Gonzales, F. (2008). GDMS: An abstraction layer to enhance Spatial Data Infrastructure usability. In *Paper presented at AGILE 2008*, Girona, Spain.

BostonGIS (2009). *Cross Compare SQL Server 2008 Spatial, PostgreSQL/PostGIS 1.3-1.4*, MySQL 5–6. <bostongis.com> (Accessed 17.01.12).

British Ordnance Survey (2010). *OS OpenSpace Beta: General FAQs*. <http://openspace.ordnancesurvey.co.uk/openspace/faq.html> (Accessed 17.01.12).

Burghardt, D., Neun, M., & Weibel, R. (2005). Generalization services on the web – classification and an initial prototype implementation. *Cartography and Geographic Information Science, 32*(4), 257–268.

Câmara, G., Vinhas, L., Modesto, Cartaxo., & de Souza, R. (2012). Free and open source GIS: Will there ever be a Geo-Linux? In E. Bocher & M. Neteler (Eds.), *Geospatial free and open source software in the 21st century* (pp. 229–246). LNG&C, Heidelberg: Springer.

Câmara, G., Vinhas, L., Reis Ferreira, K., Ribeiro De Queiroz, G., Cartaxo Modesto De Souza, R., Vieira Monteiro, M., et al. (2008). TerraLib: An open source GIS library for large-scale environmental and socio-economic application. In G. B. Hall & M. G. Leahy (Eds.), *Open source approaches in spatial data handling* (pp. 247–270). Berlin: Springer.

Cepicky, J., & Becchi, L. (2007). Geospatial processing via internet on remote servers – PyWPS. *OSGeo Journal, 1*. <http://osgeo.org/journal> (Accessed 17.01.12).

Chen, D., Shams, S., Carmona-Moreno, C., & Leone, A. (2010). Assessment of open source GIS software for water resources management in developing countries. *Journal of Hydro-environment Research, 4*, 253–264.

Christl, A. (2008). Free software and open source business models. In G. B. Hall & M. G. Leahy (Eds.), *Open source approaches in spatial data handling* (pp. 21–48). Berlin: Springer.

Conrad, O. (2007). *SAGA – Entwuf, Funktionsumfang und Anwendung eines Systems für Automatisierte Geowissenschaftliche Analysen*. PhD thesis. University of Göttingen.

Costa, G. A. O. P. C., Pinho, C. M. D., Feitosa, R. Q., Almeida, C. M., Kux, H., & Oliveira, D. A. B. (2007). *An open source environment for image interpretation in oil industry applications*. In CITARE 2007 – Congresso Íbero-Americano de Inovação Tecnológica e Áreas Estratégicas, Rio de Janeiro, Brazil, 2007. <http://www.lvc.ele.puc-rio.br/wp/?p=623> (Accessed 17.01.12).

Cox, S., Daisey, P., Lake, R., Portele, C., & Whiteside, A. (2005). *Opengis geography markup language (GML) implementation specification, version 3.1.1*. Open Geospatial Consortium.

Cruz, D., Wieland, T., & Ziegler, A. (2006). Evaluation criteria for free/open source software products based on project analysis. *Software Process Improvement and Practice, 11*, 107–122.

de la Beaujardiere, J. (2004). *OpenGIS web map service (WMS) implementation specification*. Open Geospatial Consortium.

Delmont, S. (2012). *Good Bye, Google Maps…Thanks for All the Fish*. <http://plus.google.com/u/0/118383351194421484817/posts/foj5A1fURGt> (Accessed 20.06.12).

Dickson, S. (2008). *Ordnance survey ban Google Maps*. <http://puffbox.com/2008/11/19/ordnance-survey-ban-google-map/> (Accessed 17.01.12).

Donnelly, F. P. (2010). Evaluating open source GIS for libraries. *Library Hi Tech, 28*(1), 131–151.

Duvander, A. (2010). *Map Scripting 101: An example-driven guide to building interactive maps with Bing, Yahoo!, and Google Maps*. San Francisco: No Starch Press.

Erle, S., Gibson, R., & Walsh, J. (2005). *Mapping hacks: Tips and tools for electronic cartography*. Sebastopol: O'Reilly Media.

Ertz, O., Laurent, M., Rappo, D., Sea-Tang, A., & Taillard, E. (2009). PAL – A cartographic labelling library. Position IT July 2009 56–61. <geosysin.iict.ch> (Accessed 17.01.12).

ESRI – Environmental Systems Research Institute (2012). *GIS Dictionary > desktop GIS*. <support.esri.com/en/knowledgebase/Gisdictionary/term/desktop-GIS> (Accessed 17.01.12).

Fenoy, G., Bozon, R, & Raghavan, V. (in press). ZOO-project: The open WPS platform. *Applied Geomatics.* http://dx.doi.org/10.1007/s12518-011-0070-0.

Finck, M., & Bleek, W. G. (2006). Mythen, Märchen, Missverständnisse – Eine nüchterne informatische Betrachtung von Open-Source-Entwicklungs-prozessen. In M. Bärwolff, R. A. Gehring, & B. Lutterbeck (Eds.), *Open Source Jahrbuch 2006 – Zwischen Softwareentwicklung und Gesellschaftsmodell* (pp. 207–218). Berlin: Lehmanns Media. <opensourcejahrbuch.de> (Accessed 17.01.12).

Foerster, T., Burghardt, D., Neun, M., Regnauld, N., Swan, J., & Weibel, R. (2008). Towards an interoperable web generalisation services framework – Current work in progress. In *Paper presented at 2008 ICA workshop on generalization and multiple representation*, Montpellier, France. <http://aci.ign.fr/montpellier2008/program.php> (Accessed 17.01.12).

FSF (2012a). *GNU operation system – Licenses*. <http://www.gnu.org/licenses/licenses.html> (Accessed 19.06.12).

FSF (2012b). *GNU operation system – Categories of free and nonfree software*. <http://www.gnu.org/philosophy/categories.html> (Accessed 19.06.12).

Gahegan, M., Hardisty, F., Demsar, U., & Takatsuka, M. (2008). GeoVISTA studio: Reusability by design. In G. B. Hall & M. G. Leahy (Eds.), *Open source approaches in spatial data handling* (pp. 201–220). Berlin: Springer.

Gangadharan, G. R., D'Andrea, V., De Paoli, S., & Weiss, M. (2009). Managing license compliance in free and open source software development. *Information Systems Frontiers, 12*(2), 143–154.

Google API Terms (2012). *Google Maps/Google Earth APIs terms of service*. <http://developers.google.com/maps/terms?hl=en> (Accessed 17.01.12).

Google Geo Developers Blog (2011). *Updates to the Google Maps/Google Earth APIs terms of service*. <http://googlegeodevelopers.blogspot.com/2011/04/updates-to-google-maps-apigoogle-earth.html> (Accessed 17.01.12).

Google Maps Legal Notices (2012). *Legal notices for Google Maps*. <http://google.com/help/legalnotices_maps.html> (Accessed 17.01.12).

Grassmuck, V. (2004). *Freie Software zwischen Privat- und Gemeineigentum* (vol. 458). Bonn: Bundeszentrale für politische Bildung. <http://freie-software.bpb.de/>.

Greener, S. (2009). Oracle spatial for PostGIS users – understand, isolate and migrate. In *Presentation at FOSS4G 2009 conference*, Sydney, Australia. <spatialdbadvisor.com/file_download/43/FOSS4G_2009_PostGISAndOracleSpatial.pdf> (Accessed 17.01.12).

Güting, R. H. (1994). An introduction to spatial database systems. *The VLDB Journal, 3*, 357–399.

Hall, G. B., & Leahy, M. G. (Eds.). (2008). *Open source approaches in spatial data handling*. Berlin: Springer.

Hars, A., & Ou, S. (2002). Working for free? Motivations of participating in open source projects. *International Journal of Electronic Commerce, 6*(3), 25–39.

Hazzard, E. (2011). *Open layers 2.10: Create, optimize, and deploy stunning cross-browser web maps with OpenLayers JavaScript web-mapping library – Beginner's Guide*. Birmingham: PACKT Publishing.

Hengl, T., Gruber, S., & Shrestha, P. D. (2003). *Digital terrain analysis in ILWIS – lecture notes and user guide*. <www.itc.nl/personal/Shrestha/DTA> (Accessed 17.01.12).

Herring, J. R. (Ed.) (2010). *OpenGIS implementation specification for geographic information – Simple feature access – Part 2: SQL option*. Open Geospatial Consortium.

Herring, J. R. (Ed.) (2011). *OpenGIS implementation specification for geographic information – Simple feature access – Part 1: Common architecture*. Open Geospatial Consortium.

Hugentobler, M. (2008). Quantum GIS. In S. Shekhar & H. Xiong (Eds.), *Encyclopedia of GIS* (pp. 935–939). New York: Springer.

Ince, D. C., Hatton, L., & Graham-Cumming, J. (2012). The case for open computer programs. *Nature, 482*(7386), 485–488.

ISO 19128 (2005). *Geographic information – Web map service interface*. Geneva, Switzerland: International Organization for Standardization.

ISO/DIS 19142 (2009). *Geographic information – Web feature service*. Geneva, Switzerland: International Organization for Standardization.

Jensen, J. R. (2006). *Remote sensing of the environment: An earth resource perspective* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Johnson, K. (1999). *Open-source software development*. <http://chinese-school.netfirms.com/computer-article-open-source.html> (Accessed 17.01.12).

Jolma, A., Ames, D. P., Horning, N., Mitasova, H., Neteler, M., Racicot, A., et al. (2008a). Free and open source geospatial tools for environmental modelling and management. In A. J. Jakeman, A. A. Voinov, A. E. Rizzoli, & H. C. Serena (Eds.), *Environmental modelling, software and decision support* (pp. 163–180). Amsterdam: Elsevier.

Jolma, A., Ames, D. P., Horning, N., Mitasova, H., Neteler, M., Racicot, A., et al. (2008b). Environmental modeling using open source tools. In S. Shekhar & H. Xiong (Eds.), *Encyclopedia of GIS* (pp. 275–279). New York: Springer.

Kropla, B. (2005). *Beginning MapServer: Open source GIS development*. Berkeley, CA: Apress.

Lupp, M. (2007). *Styled layer descriptor profile of the web map service implementation specification*. Open Geospatial Consortium.

Lupp, M. (2008). Deegree free software. In S. Shekhar & H. Xiong (Eds.), *Encyclopedia of GIS* (pp. 235–239). New York: Springer.

Martinez-Llario, J., & Gonzalez-Alcaide, M. (2011). Design of a Java spatial extension for relational databases. *Journal of Systems and Software, 84*(12), 2314–2323.

Martin-Reinhold, C., Huerta, J., & Granell, C. (2010). gvSIGDroid – An open source Gis solution for the Android platform. In *Proceedings of WEBIST 2010*, Valencia.

Microsoft (2012). *Bing developer center: Terms of use*. <http://www.bing.com/developers/tou.aspx> (Accessed 17.01.12).

Milosavljević, A., Dordevic-Kjan, S., & Stoimenov, L. (2008). An application framework for rapid development for web based GIS: Ginis Web. In J. T. Sample, K. Shaw, S. Tu, & M. Abdelguerfi (Eds.), *Geospatial services and applications for the Internet* (pp. 49–72). New York: Springer.

Mische, V. (2011). The state of GeoCouch. In *Presentation at FOSS4G 2011 conference*, Denver, USA. <http://vmx.cx> (Accessed 28.06.12).

Montesinos, M., & Carrasco, J. (2010). A comparison of GIS mobile applications. In *Presentation at FOSS4G 2010 conference*, Barcelona, Spain. <http://www.slide-share.net/migmontesinos/comparison-of-mobile-gis-applications> (Accessed 17.01.12).

Morin, A., Urban, J., Adams, P. D., Foster, I., Sali, A., Baker, D., et al. (2012). Shining light into black boxes. *Science, 336*(6078), 159–160.

Mota, G. L. A., Nunes e Silva Brito, J. L., Araujo Ribeiro, J., Bernardo Filho, O., Teixeira Silveira, M., Alves de Aguiar, R., et al. (2012). The e-foto project and the research to implement a GNU/GPL open source educational digital photogrammetric workstation? In E. Bocher & M. Neteler (Eds.), *Geospatial free and open source software in the 21st century* (pp. 89–106). Heidelberg, Springer: LNG&C.

Nagy, D., Yassin, A. M., & Bhattacherjee, A. (2010). Organizational adoption of open source software. *Communications of the ACM, 53*(3), 148–151.

Neteler, M., & Raghavan, V. (2006). Advances in free software geographic information systems. *Journal of Informatics, 3*(2). <http://creativecity.gscc.osaka-cu.ac.jp/JI/viewarticle.php?id=36&layout=abstract> (Accessed 28.05.12).

Neteler, M., Bowman, M. H., Landa, M., & Metz, M. (2012). GRASS GIS: A multi-purpose open source GIS. *Environmental Modelling & Software, 31*(May), 124–130.

Neteler, M., & Mitasova, H. (2008). *Open source GIS: A GRASS GIS approach* (3rd ed.). Berlin: Springer.

Neun, M., Burghardt, D., & Weibel, R. (2008). Web service approaches for providing enriched data structures to generalisation operators. *International Journal of Geographical Information Science, 22*(2), 133–165.

Obe, R. O., & Hsu, L. S. (2010). *Compare SQL Server 2008 R2, Oracle 11G R2, Postgre SQL/PostGIS 1.5 spatial features*. <http://bostongis.com/PrinterFriendly.aspx?content_name=sqlserver2008r2_oracle11gr2_postgis15_compare> (Accessed 17.01.12).

Obe, R. O., & Hsu, L. S. (2011). *PostGIS in action*. Greenwich: Manning.

Olaya, V. (2004). *A gentle introduction to SAGA GIS*. <http://downloads.sourceforge.net/saga-gis/SagaManual.pdf>.

Orchard, L., Pehlivanain, A., Koon, S., & Jones, H. (2009). *Professional JavaScript Frameworks: Prototype, YUI, ExtJS, Dojo and MooTools*. Birmingham: Wrox Press Ltd..

Pahl, M. (2003). *Architektur eines wissensbasierten Systems zur Interpretation multisensorieller Fernerkundungsdaten*. PhD thesis. University of Hannover. <www.tnt.uni-hannover.de/~pahl/diss.pdf> (Accessed 28.05.12).

Parsons, E. (2008). *Who reads the terms of service anyway…*. <http://edparsons.com/2008/11/who-reads-the-terms-of-service-anyway/> (Accessed 17.01.12).

Perens, B. (2005). *The emerging economic paradigm of open source*. First Monday, Special Issue #2.

Pieper Espada, G. (2008). Free and open source software for land administration systems: A hidden treasure? FIG working week 2008, Stockholm, Sweden. <http://fig.net/pub/fig2008/papers/ts03c/ts03c_03_pieperespada_2711.pdf> (Accessed 17.01.12).

Pintscher, L. (Ed.) (2012). *Open advice – FOSS: What we wish we had known when we started*. <http://open-advice.org> (Accessed 02.07.12).

Ramsey, P. (2006). uDig desktop application framework. In *Presentation at FOSS4G 2006 conference*, Lausanne, Switzerland. <http://2006.foss4g.org/contributionDisplay74db.html?contribId=132&sessionId=44&confId=1> (Accessed 17.01.12).

Ramsey, P. (2007). The state of open source GIS. In *Presentation at FOSS4G 2007 conference*, Vancouver, BC, Canada. <http://refractions.net/expertise/whitepapers/> (Accessed 17.01.12).

Ramsey, P. (2009). *Open sources #16 – Open for business* (pp. 34–36). GeoConnections International (December/January 2009).

Ramsey, P. (2011). The state of PostGIS. In *Presentation at FOSS4G 2011 conference*, Denver, USA. <http://s3.cleverelephant.ca/state-of-postgis-2011-foss4g.pdf> (Accessed 17.01.12).

Reid, J., & Martin, F. (2001). The open source movement and its potential in implementing Spatial Data Infrastructures. In *International symposium on spatial data infrastructure*, Melbourne, Australia. <http://sourceforge.net/projects/mentaer.u/files/pubs_by_others/reid_martin_foss_for_sdi_2001.pdf> (Accessed 17.01.12).

Rey, S. J. (2009). Show me the code: Spatial analysis and open source. *Journal of Geographical Systems, 11*(2), 191–207.

Rey, S. J., & Anselin, L. (2007). PySAL: A Python library of spatial analytical methods. *The Review of Regional Studies, 37*(1), 7–27.

Rey, S. J., & Janikas, M. V. (2006). STARS: Space–time analysis of regional systems. *Geographical Analysis, 38*, 67–84.

Rocchini, D., & Neteler, M. (2012). Let the four freedoms paradigm apply to ecology. *Trends in Ecology & Evolution, 27*(6), 310–311.

Rogers, E. M. (1995). *Diffusion of innovations*. Glencoe: Free Press.

Samet, H. (2006). *Foundations of multidimensional and metric data structures*. San Francisco, CA: Morgan Kaufmann.

Sanz-Salinas, J.-C., & Montesinos-Lajara, M. (2009). Current panorama of the FOSS4G ecosystem. *Upgrade, 10*(2), 43–51.

Schaeffer, B., & Foerster, T. (2008). A client for distributed geo-processing and workflow. *Journal of Location Based Services, 2*(3), 194–210.

Schut, P. (Ed.) (2007). *OpenGIS Web processing service*. Open Geospatial Consortium.

Sherman, G. E. (2012). *The geospatial desktop GIS: Open source GIS and mapping*. Locate Press.

Simonetto, E., & Follin, J.-M. (2012). An overview on interferometric SAR software and a comparison between DORIS and SARSCAPE packages. In E. Bocher & M. Neteler (Eds.), *Geospatial free and open source software in the 21st century* (pp. 107–122). LNG&C, Heidelberg: Springer.

Slingsby, A., Dykes, J., & Wood, J. (2009). Configuring hierarchical layouts to address research questions. *IEEE Transactions on Visualization and Computer Graphics, 15*(6), 977–984.

Stallman, R. M. (1999). The GNU operating system and the free software movement. In C. DiBona, S. Ockman, & M. Stone (Eds.), *Open sources: Voices from the open source revolution* (pp. 53–50). Sebastopol: O'Reilly.

Stallman, R. M. (2009). Why "open source" misses the point of free software. *Communications of the ACM, 52*(6), 31–33.

Steiniger, S., & Michaud, M. (2009). The Desktop GIS OpenJUMP: A hands-on introduction. In *OGRS 2009 workshop*, Nantes, France. <http://sourceforge.net/projects/jump-pilot/files/Documentation/> (Accessed 17.01.12).

Steiniger, S., & Weibel, R. (2009). *GIS software – a description in 1000 words*. <http://sourceforge.net/projects/mentaer.u/files/pubs/gissoftware_steiniger2008.pdf> (Accessed 17.01.12).

Steiniger, S., & Bocher, E. (2009). An overview on current free and open source desktop GIS developments. *International Journal of Geographical Information Science, 23*(10), 1345–1370.

Steiniger, S., & Hay, G. J. (2009). Free and open source geographic information tools for landscape ecology. *Ecological Informatics, 4*(4), 183–195.

Steiniger, S., & Hunter, A. J. S. (2012). Free and open source GIS software for building a spatial data infrastructure. In E. Bocher & M. Neteler (Eds.), *Geospatial free and open source software in the 21st century* (pp. 247–261). LNG&C, Heidelberg: Springer.

Sveen, A. F. (2008). *Use of free and open source GIS in commercial firms*. Master thesis. Norwegian University of Technology and Science. <http://code.atlefren.net/download/dl.php?id=10> (Accessed 17.01.12).

Takatsuka, M., & Gahegan, M. (2002). GeoVISTA studio: A codeless visual programming environment for geoscientific data analysis and visualization. *Computers an Geosciences, 28*, 1131–1141.

Turton, I. (2008). Geo tools. In G. B. Hall & M. G. Leahy (Eds.), *Open source approaches in spatial data handling* (pp. 153–169). Berlin: Springer.

Tweedie, C. (2009). *Image server benchmark*. <http://blog.webmapper.com.au/image-server-benchmark/> and <http://sourceforge.net/projects/mentaer.u/files/pubs_by_others/tweedie_Image_Server_Benchmark_2009.pdf> (Accessed 17.01.12).

Valenzuela, C. R. (1988). ILWIS overview. *ITC Journal, 1*, 4–14.

Ven, K., Verelst, J., & Mannaert, H. (2008). Should you adopt open source software? *IEEE Software, 25*(3), 54–59.

Vretanos, P. A. (Ed.) (2005a). *Web feature service implementation specification*. Open Geospatial Consortium.

Vretanos, P. A. (Ed.) (2005b). *OpenGIS filter encoding implementation specification*. Open Geospatial Consortium.

Wagner, J. -O. (2006). Quo Vadis Desktop-GIS? In *Presentation at Quo Vadis GIS workshop*, WhereGroup, 21, Bonn, Germany (December 2006). <http://www.wheregroup.com/files/DesktopGIS.pdf> (Accessed 17.01.12).

Wheeler, D. A. (2007). *Why open source software/free software (OSS/FS, FLOSS, or FOSS)? Look at the numbers!*. <http://www.dwheeler.com/oss_fs_why.html> (Accessed 17.01.12).

Please cite this article in press as: Steiniger, S., & Hunter, A. J. S. The 2012 free and open source GIS software map – A guide to facilitate research, development, and adoption. *Computers, Environment and Urban Systems* (2012), http://dx.doi.org/10.1016/j.compenvurbsys.2012.10.003