CONCEPT, DESIGN AND DEVELOPMENT OF LAND ADMINISTRATION AND CADASTRE SYSTEMS USING OPEN SOURCE SOFTWARE

TAREK ZEIN, CHRISTIAN TIMM and DR PAUL HARTFIEL
Hansa Luftbild AG, Germany
zein@hansaluftbild.de

Paper prepared for presentation at the
“2015 WORLD BANK CONFERENCE ON LAND AND POVERTY”

Copyright 2015 by author(s). All rights reserved. Readers may make verbatim copies of this document for non-commercial purposes by any means, provided that this copyright notice appears on all such copies.
Abstract
In deploying new or modernizing existing land administration and cadastre systems the relevant authorities / agencies must decide whether to use open source or proprietary software. Neither open source nor proprietary software can fully cater to the requirements of land administration and cadastre authorities. The two types of platform will require customization in order to meet the requirements of the requesting authority, region or country. Such requirements will vary according to land legislation and local/regional regulations. Cadastre and land administration systems in developed countries have been established over a long period of time and these are usually implemented using expensive proprietary solutions. In most developing countries land administration systems only exist in fragments. Proprietary software solutions implemented in developed countries cannot be ported to developing countries since the system and administrative incompatibilities are too great. This paper discusses the advantages of open source software (OSS), contrasting it with proprietary software. It then goes on to describe the building of land registration and cadastre systems based on open source software and proposes a reference architecture for such systems based on the generic ISO Land Administration Domain Model (LADM). The paper also presents an overview of existing open source projects which are relevant to land administration and cadastre.

Key Words:
cadastre systems, document management system, land administration, open source software, proprietary software, spatial data infrastructure, Land Administration Domain Model
Introduction

In deploying new, or modernizing existing land administration and cadastre systems (LAS) the relevant authorities / agencies are often confronted with an important decision - should they use open source or proprietary software? Neither option can fully cater to the requirements of land administration and cadastre authorities. Both types of platform will require customization in order to meet the requirements of the requesting authority, region or country.

Land legislation is crucial to the economic and political stability of any country. Developed and developing nations have taken different trajectories in the development of their land administration systems. Developed countries have established their land administration systems over many decades. Computerization of these complex systems was frequently undertaken using proprietary software. Developing countries have developed their land administration systems in response to demand and need. This has meant that the technology applications employed and the organizational / legal frameworks applied have lacked overall consistency. In addition technology applications introduced on an ad hoc basis have significantly impacted the nations' legal and organizational frameworks, which in turn have had an influence on the next round of technological innovations. Due to these different trajectories land administration proprietary software solutions implemented in developed countries cannot be ported to developing countries. The system and administrative incompatibilities are too great.

This paper will discuss and describe the application of OSS to land administration. Initially it will outline the advantages of OSS in comparison to proprietary software, examining the worldwide utilisation of OSS. Following this the functionalities required of land administration systems will be outlined and the ISO standard Land Administration Model (LADM) described. A reference architecture based on a generic LADM will then be proposed. Finally the paper will present an overview of existing open source projects which are relevant to land administration and cadastre, including the software package Solutions for Open Land Administration (SOLA).

Open Source Versus Proprietary Software

Many online discussions and blogs address the issue of the relative advantages of open source based solutions versus those of proprietary based solutions.

Kepes (2013), for example, claims that proprietary software is a clear winner because businesses still favour it. Wheeler (2014) states that OSS “has significant market share in many markets, is often the most reliable software and in many cases has the best performance”. He also states that the governments in many countries have adopted policies regarding the use of OSS. However, Backaitis (2013) suggests that
neither approach has emerged with a clear advantage. The claimants bolster their arguments with evidence based studies, facts, and statistics. Regardless of the opinion of these writers and bloggers, open source has become a major player in the software industry. In addition open source is now utilised by proprietary software developers to enhance their products, and to develop new products.

The motivation of proprietary software companies is to make a profit. The commercial foundation of proprietary software is the licence. The more licences sold the more profit the software company makes. Proprietary software companies are under pressure to continually upgrade their products and to sell users upgrades and new versions. Whenever there is a new software release, customized applications have to be updated (against payment). Customization of the software to fit the entire range of user needs is undertaken by the software company itself or by business partners of the company. This often means that detailed knowledge of the system is confined to a limited number of IT professionals. The user must revert to the software company or its partners for modifications and extensions to their software system. In order to protect this business model, software companies often restrict access to the application programming interface (API) and the data structures. Thus customers are severely limited in their choice of IT specialist/support services and/or the opportunity to undertake their own development. This can be a major problem when system migration or modification is required.

By contrast the open source community is driven not by the profit motive but by the need to solve problems. Its business model is not based on licensing, but on services and solutions. This does not mean that the development of open source software is free of commercial interests or of costs; there are companies who sell open source development and support services. Nevertheless, OSS allows users and developers free usage and access. In the past programs developed by the open source community were aimed at the proficient user. The software was generally complicated and inaccessible to the average user. Documentation was sparse and software training and support were not easily available. This situation has changed. Organisations or communities now tailor and customize once unfriendly software to suit end users of all knowledge levels. In addition there are now companies which offer support and training in OSS development, so that the source code can be read and modified by all. The code for any large project can be scrutinised by many more developers than even the biggest software companies can employ. This allows software flaws to be discovered early in the development process rather than when the system has gone live. Most open source projects allow anyone to contribute, and problems are normally resolved quickly and cleanly. Knowledge is shared, not monopolised by a few as is the case with proprietary software.
So what is OSS? In general OSS is the opposite of closed or proprietary software but not the opposite of commercial software, since OSS is often developed with commercial interests in mind. Many definitions of OSS exist. One such definition is “Open-source software (OSS) is computer software with its source code made available with a license in which the copyright holder provides the rights to study, change and distribute the software to anyone and for any purpose.” (St. Laurent, 2008, pp. 4). The key word in this definition is license. There are several license types for OSS. The best known are:

- GNU General Public Licence (GPL) (FSF, n.d.)
- Apache License (Apache, n.d.)
- MIT license (MIT, n.d.)

All these licenses have features in common, which form the core characteristics of OSS. These features are:

- The source code is available, can/must be distributed, and can be modified
- No licence fees are charged

Although OSS does not charge for software usage (licensing costs), costs may still be incurred by the end-user. These will be the result of software tailoring, and services such as support and training.

How is OSS developed? The classic approach as described by Raymond (1999) in *The Cathedral and the Bazaar* is that of a collaborative one. This involves many professionals pooling skills and ideas to develop a piece of OSS. A particularly famous example of this approach is that of the Linux Kernel.

In addition to this classic approach OSS is also often developed by commercial companies and then released under an OSS license. Typical of such an approach is the browser Google Chrome. Google developed Chrome, because it needed a stable and fast platform to support the use of its online services.

Open source software is utilised by both international and national organisations. For example the US Department of Homeland Security has developed its own intrusion detection system called Suricata – a complete suite based on open source software (Evans, 2012).

A further example of the acceptance of the OSS by major organisations is that of the uptake of the open source development PostgreSQL. This enterprise class, powerful, open source object-relational database management system, was actively developed over more than 15 years, and is reputed for its proven architecture, reliability, data integrity, and accuracy. It is used by a wide number of organizations operating in sectors such as education, technology, media and government. Government sector users include the United States of America State Department (http://www.postgresql.org/about/users).
Many major organisations have established formal policies with regard to the use of OSS. In 2000 the European Commission (EC) defined its strategy concerning the internal use of open source software. Figure 1 charts software utilized by the EC in 2000. As can be seen all the software is commercial. Figure 2 charts software utilized by the EC in 2014. It can be seen that OSS has become a major component of the EC’s software base.

Figure 1: Software solutions utilized by the European Commission in the year 2000 (source Damas, Directorate-General for Informatics (DIGIT) of the European Commission, 2014)
Two of the EC’s key strategic components for the internal use of open source software are:

1. to continue to adopt formally (through the Product Management procedure) the use of open source software technologies and products where a clear benefit can be expected.
2. to consider open source software solutions alongside proprietary ones in IT procurements. Contracts are awarded on a "value for money" basis.

(http://ec.europa.eu/dgs/informatics/oss_tech/index_en.htm)

**Land Administration Systems**

Chartered rights on land are very important for the economic and political stability of a country. In many developing countries land administration systems (LAS) only exist in fragments. This is not only with regard to implemented technical IT solutions but also with regard to the organisational and legal aspects of the land administration system.

Land administration is defined as “…the processes of recording and disseminating information about the ownership, value and use of land and its associated resources.” (UN-ECE, 1996). A LAS must support
four main land management functions: land tenure, land value, land use and land development (Williamson, Enemark, Wallace, Rajabifard, 2010, pp. 119-120). This means that all the relevant legal, social and technical parameters must be in place, if an LAS is to be implemented.

The modernisation of any land administration sector in developing countries often starts with a project to implement a technical system for cadastre and property registration. At this stage the legal or organisational foundation is often not in place. Therefore the implementation of a computerised land administration system will lead to substantial changes in the legal and organisational framework, which in its turn will have a major impact on the design of the technical solution. Any purely technical IT system solution will fail if local constraints and parameters are not thoroughly understood and taken into account.

No two countries will ever implement the same land administration system; even when a widely accepted LAS standard, such as the ISO 19512 Land Administration Domain Model (LADM), is applied. The cultural, historical and social backgrounds, of a country, have a significant impact on the structure and the design of the land administration system. For example the answer to “what is ownership?” will reveal if in the relevant country there is a concept of private ownership, if there are specific traditional rights and how a parcel is defined.

Williamson et al. (2010, pp. 6) state in their book *Land Administration for Sustainable Development* that four basic ingredients are involved in the design of a national LAS:

- The land management paradigm with its core administration functions
- The common processes found in every system
- A toolbox approach offering tools and implementation options
- A role for land administration in supporting sustainable development

The land management paradigm is the basic framework for setting up an LAS. It is a theoretical and conceptual model. It helps organisations to design, construct and monitor an LAS, at the technical, organisational, social and legal levels. The land management paradigm covers the four main functions of land management: land tenure, land value, land use and land development (Williamson et al., 2010, pp. 119-120).

The second ingredient is that of the common processes. Such processes will be country specific and typically define parameters such as dividing up land, tracking changes or distributing land. They will differ in details from country to country, but they have to exist in all LASs.
With regard to the third ingredient, the toolbox approach refers to those tools which countries implement that “reflect on the capacity and the history of a country”. Some tools can be general such as land policies or land markets, while other tools can be professional relating to tenure, registration systems, boundaries, etc..

Williamson’s fourth ingredient is the land administration sector’s role in sustainable development. Sustainable development is managed by invoking the land management paradigm to determine those tools to be used / necessary for handling environmental, economic or social processes.

**Land Administration Domain Model (LADM)**

Modern LAS implementations frequently take as a reference the ISO standard Land Administration Domain Model (LADM) (ISO 19152:2012). This standard is the result of wide-ranging consultations between many international experts; thus, its use in developing an LAS ensures access to a common terminology vis-à-vis land administration systems and their computerization.

The ISO LADM has two main goals:

- providing an extensible basis for the development and refinement of efficient and effective land administration systems, based on a Model Driven Architecture (MDA), and
- enabling the involved parties, both within one country and between different countries, to communicate, on the basis of the shared vocabulary (that is, an ontology) implied by the model.

Therefore the ISO LADM is a conceptual framework for the static part of an information system, and for interfaces. Furthermore it provides an ontology as a foundation for information exchange and interoperability. LADM is not a data model, however, a concrete data model can be derived from it.

The ISO LADM defines structures for the following phenomena:

- parties (people and organizations);
- basic administrative units, rights, responsibilities, and restrictions (ownership rights);
- spatial units (parcels, buildings and networks);
- spatial sources (surveying / mapping); and
- spatial representations (geometry and topology).

Figure 3 shows an overview of all parts of the ISO LADM and their relationship and Figure 4 illustrates the core idea of the LADM.
Figure 3: Overview of the conceptual schema of the ISO LADM (Source: International Organization for Standardization, 2012)

Figure 4: Basic classes of ISO LADM (Source: International Organization for Standardization, 2012)

The diagram in Figure 4 can be read in the following way (for parcels):

- Party (an individual person or legal person) is modelled as LA_Party
• A parcel modelled as an LA_SpatialUnit. This is the spatial representation of a parcel in the cadastre. This is the more technical way of looking at a parcel
• A parcel is covered by the legal representation of the parcel. This is modelled as an LA_BAUnit.
• Parcels and parties are independent objects
• A parcel and a party is connected by one or more of right, restriction or responsibility. This is modelled in LA_RRR. A typical right is e.g. ownership. Depending on the concrete modelling, different type of rights, restrictions or responsibilities are available. This depends on historical, traditional, social and legal requirements.

In 2011 the ISO LADM formed the basis for the development of a real property registration system for the city of Addis Ababa (Zein, Hartfiel, Berisso, 2012). It was used as the base conceptual model and a two-step development process was followed. First the conceptual model of the land administration domain was developed. This was followed by the development of the database implementation. The data model was optimized mainly for a production database for which data capture and editing are the most common and most important operations. Consequently, the design emphasis was on consistency and stability rather than query performance. To achieve this the data model was kept free of redundancies and wherever possible the relationships between objects were modelled with foreign key constraints. This development used the Oracle database management system. The principal implementation and modelling techniques can, however, be transferred to any relational database with a spatial storage extension.

Using Open Source Software in LAS Development
Today OSS plays a major role in the world-wide IT landscape and is used by both private users and by governmental and private organisations. Important parts of the internet infrastructure are driven by OSS, the Internet’s Domain Name System (DNS) is mostly implemented on the OSS BIND; most web servers are operated by APACHE (http://w3techs.com/technologies/overview/web_server/all); Google Chrome is the most used web browser (http://en.wikipedia.org/wiki/Usage_share_of_web_browsers). All these programs have several features in common
• open source products,
• professional,
• innovative,
• development driven by / sponsored by large organizations, and
• tailored to support higher level business models

The last feature is the most important one, especially in the context of implementing an LAS. The main purpose of an LAS is not to build a repository of parcels, land records and tenures but to support the land
development, planning and taxation. Cadastre and land registration systems are the tools which support these land processes. Regardless of its legal, social or historical characteristics the core of an LAS is always its cadastre and land registration system.

Due to their commonalties, cadastre and land registration systems can be developed collaboratively by any interested community and improvements can be shared and used by all. The tailoring of a concrete LAS system can be done as an open source project or as a closed development, depending on the needs of the LAS operator. The collaborative approach can significantly reduce the total cost of ownership of an LAS.

If the OSS concept is transferred to the LAS domain, the technical core building blocks of the LAS, which are the cadastre and land register, can be viewed in a similar way to Chrome, in that they support high land administration processes. The cadastre and land register are the tools for driving the entire LAS system. They are necessary from a business point of view but not sufficient, because an LAS should deliver information / processes required for land administration.

The technical implementation of an LAS can be seen as an onion layer model as is illustrated in Figure 5.

Figure 5: Layered model of the technical implementation of an LAS (Source: Hansa Luftbild, 2015)
The inner layer, the IT layer, is the general purpose IT system which is completely domain independent. The main components of this layer are the database and geographic information systems (GIS).

The next layer, the cadastre and land register layer, contains the cadastre and land registration systems. In this layer both systems provide the general functionality common to all LASs.

The layer above, the LAS layer, contains the implementation which is specific to a concrete LAS. This includes the processes and models.

The last layer, the spatial data infrastructure (SDI) layer, contains the interface to services and systems based on land information and to spatially enabled government processes thus ensuring a sustainable development.

The IT layer and the cadastre and land register layer are independent from a concrete LAS. Therefore these layers are ideal candidates for an OSS implementation based on the collaborative approach (the bazaar model).

**Existing OSS for LAS**

The OSS community has already created and developed a useful number of software solutions which can be used to build a complete LAS. Many open source components are available to implement the IT layer. This layer provides the core domain independent functionalities of:

- (spatial) data persistence, and
- spatial data handling

In addition the IT-Layer has to provide other services, such as authentication, transaction processing, and general business logic. These are however features which are common to all complex IT systems.

Data persistence functionality can be implemented using the OSS PostgreSQL in combination with PostGIS. The latter is an extension for handling spatial data. PostgreSQL is a state of the art object-relational database. It provides multiple paradigms:

- The standard relational paradigm. PostgreSQL implements the majority of the ISO SQL:2011 standard with full ACID (Atomicity, Consistency, Isolation, Durability) support and the MVCC (multiversion concurrency control) transaction mechanism.
- A mechanism for implementing user defined data types. This includes operations, methods and indexes which constitute features of the object relational extensions
- NoSQL features, like JASON handling and Key-Value data stores
The PostGIS extension provides a fully OGC compliant mechanism to store, retrieve and process spatial data, with close integration into the SQL languages of PostgreSQL. PostGIS now also provides topological storage of data. This feature is especially useful when highly accurate spatial data such as cadastral data is needed.

For the other core function of the IT layer spatial data handling by the GIS; there are also state of the art OSS systems available, including the desktop based solution QGIS (www.qgis.org). QGIS is already used by several cadastral and public administrations, for example the Swiss Canton of Solothurn has introduced QGIS with the intention to free itself from proprietary GIS software (Duester, 2008).

In order to provide web services an LAS needs web based tools. These are also available as OSS components. All web based systems should use OGC standards for interoperability. The state-of-the-art OSS server implementation of OGC services is GeoServer (geoserver.org). GeoServer supports all important and relevant OGC standards including:

- Web Map Service (WMS)
- Web Feature Service (WFS). The WFS implementation includes also WFS-T (T=Transactional) which allows data updating via Web Application
- Web Coverage Service (WCS)
- Web Processing Service (WPS)

The client part of the web application can be implemented by OpenLayers, which is an OSS library for implementing OGC compliant application.

All of the above mentioned OSS components (PostgreSQL, PostGIS, QGIS, GeoServer, and OpenLayers) were used in the development of the web application ExperMaps (www.expermaps.com), a platform used for planning and land use projects in Germany, Croatia and Ethiopia. The application is currently being extended to cover a cadastre and land registration system.

A particularly good OSS reference for spatial data systems is the Open Source Geospatial Foundation (OSGeo, www.osgeo.org), a collaborative development platform for open source geospatial software.

**Proposed Reference Architecture of an LAS**

The development of a complex system such as the cadastre and land registration systems requires a structured approach. Even in the “Bazaar” development model, there is a need for a development blueprint that states the goal of the project and for a maintainer who keeps track of changes and developments, in other words a reference architecture.
Software architecture is influenced by several factors. For the architecture which will be proposed the most important factors are listed in Table 1.

<table>
<thead>
<tr>
<th>Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completely based on open source software</td>
<td>All 3rd party components (such as libraries, databases, etc.) must be published under an open source license. The software must be able to operate in a completely open source environment, including the operating system.</td>
</tr>
<tr>
<td>Reduction of complexity</td>
<td>Complexity must be reduced during the development and during operation.</td>
</tr>
<tr>
<td>Scalability</td>
<td>The system should be scalable from local to national authorities. Scaling must be horizontal and/or vertical.</td>
</tr>
<tr>
<td>Extensibility</td>
<td>A system cannot cover all requirements. Therefore the design must be extendable; either through tailoring or replacement.</td>
</tr>
</tbody>
</table>

Table 1: Important factors that influence software architecture (Source: Hansa Luftbild, 2015)

Figure 6 shows a high level overview of the proposed system and its components.
Figure 6: Diagram of a high level overview of the proposed system and its components (Source: Hansa Luftbild, 2015).

Two general design principles are applied:

- a layer design pattern, and
- a separation of concerns and high cohesion
Both principles are used to structure the system into parts or components. These parts can be equipped with well-defined interfaces and behaviours. This allows the separate development of components and the replacement of components, if and when required.

The system is structured into four main levels:

1. ExperMaps Data Level
2. Internal OGC Service Level
3. ExperMaps LAS
4. Desktop GIS - QGIS

The components of the levels are ordered according to layers, with each layer consuming only the services provided by the layer below it. The lowest layer starts with ExperMaps Data Level and the highest is the desktop GIS - QGIS.

1. ExperMaps Data Level

The data level provides the persistence services for the entire system. The level is implemented in the PostgreSQL / PostGIS database. The database hosts an ISO LADM compliant data model. This data model is, by default, not tailored for national requirements such as specific rights, restrictions or responsibilities. The data model implements the generic structure of the ISO LADM and structures which are to be tailored such as look-up tables and specialisation by inheritance.

Another part on the ExperMaps Data Level is the document management system (DMS) which is used for storing all types of cadastre and land record documents, such as sketches, field books or deeds.

There are several OSS DMSs available eg Alfresco (http://www.alfresco.com) or Agorum (http://www.agorum.com/)). Any DMS which provides a WebDAV interface can be used. The document index is stored in the PostgreSQL database and is identical for all DMSs.

2. Internal OGC Service Level

The internal OGC provides OGC compliant services. It is implemented on a GeoServer. The OGC service level is used to integrate data from external sources, such as the planning or environmental authority or organisation.

3. ExperMaps LAS

ExperMaps LAS is the core component of the system. This includes all the domain specific functionalities. This component itself is separated into two parts, the application level and the graphic user interface. The application level hosts the general business logic of the system. It uses node.js
(http://nodejs.org/), a platform for server side JavaScript, as a runtime environment. One of the major goals of node.js is scalability (http://en.wikipedia.org/wiki/Node.js ). Node.js was chosen for several reasons:

- Scalability and performance
- Ease of deployment
- Single language approach. All development in the ExperMaps LAS component can be carried out in JavaScript
- A management package which provides a range of additional functionalities

The functions inside the application level are implemented as services and are accessible via the web service interfaces. The services are grouped according to their theme. The LADM services provide, for instance, all functions which are necessary for accessing the data model. This includes the tailoring of the data model to specific national needs. The workflow service provides the functions to manage cadastre and land registration related tasks, for example creating a cadastre task such as splitting a parcel and keeping track of the workflow. All changes to the underlying data level are performed via a task within the workflow service. There are additional services for managing the archive in the DMS or ensuring security by authentication and access control. The application level can be extended to include services not mentioned yet. The existing services can provide interfaces to integrate 3rd party components.

The ExperMaps LAS graphic user interface (GUI) contains all the components which are necessary for the user interaction with the system. This is the front end of the system. The main component is the ExperMaps core client which implements the cadastre and the property registration tools built in the land registration system.

The ExperMaps core client is based on OpenLayers (http://openlayers.org/). The implementation is carried out in JavaScript. It is a full featured web mapping client with additional functionalities for cadastral data handling, such as splitting and merging parcels, according to defined cadastre rules. The ExperMaps core client operates on the SpatialUnit schema structures of the LADM implementation in the database which is accessible by the LADM connector service in the underlying application layer. The property registration tools provide access to the administrative schema and the party schema structures of the LADM implementation in the database. This includes the LA_BAUnit and the LA_RRR. The component is used for tasks such as registration of ownership, transfer of ownership or title generation.

4. Desktop GIS - QGIS

The final main component of the system is the desktop GIS. QGIS is chosen for advanced GIS operations, such as cartography, in-depth spatial analysis, importing and edit of surveying data, etc. QGIS provides
most of these functionalities either out of the box or with existing add-ons. The data transfer between QGIS and ExperMaps LAS will be performed via standard interfaces, such as the Web Feature Service (WFS) or the Geographic Markup Language (GML).

These four above-mentioned components incorporate all the parts required to build an LAS and can be implemented with existing OSS developments.

Existing LAS Developed with OSS

In both developed and developing countries, many national / regional authorities have used open source software to develop land administration systems or to establish spatial data infrastructures. One good example of such a development is the Solutions for Open Land Administration - SOLA.

In June 2010 the United Nations Food and Agriculture Organisation (FAO) instigated its SOLA project. The project, which was funded by the Government of Finland, aimed to develop a computerized cadastre and registration system based on open source software. SOLA’s implementation was based on a definition of generic cadastre and registration processes and the software requirements.

Two constraints were applied to the SOLA development

1. Open source software to be used in the implementation
2. The system should be portable to a wide range of major hardware and operating system platforms.

Taking these two constraints into consideration the outcome of the processes’ definition and requirements was an architecture based on a modular generic core making the software open to customization according to the needs of land and cadastre authorities or agencies in different countries.

The generic core included all essential functionalities of a land administration system. This core provided adaptation application programming interfaces (APIs) and extension mechanisms which allowed SOLA to be augmented and customized to the specifics of three pilot installations, in Ghana, in Samoa and in Nepal. In addition to the system development, the customization of SOLA in these countries constituted an important part of the SOLA project.

Figure 7 illustrates the architecture of SOLA which has a single extensible core. The core runs an application server that deploys system services. Example core services include a Spatial component for managing land parcels, Cadastre for handling land administration data, and Hibernate for data persistence. In addition, the core implements security related services such as auditing, authorization, and authentication. The SOLA core provides three extension points: a data extension point for supporting databases, a presentation extension point for porting to client platforms, and a services extension point for including extra services.
SOLA’s database design is based on the Land Administration Model (LADM) as laid down in ISO 19152.

During the design phase of SOLA a number of specified analysis mechanisms were identified and mapped to design mechanisms. The design mechanisms were in turn mapped to implementation mechanisms which defined the software or framework or formats to be used in the implementation of SOLA. All defined software and frameworks fell under the open source software licenses (GPL or LGPL). Table 2 shows how the analysis mechanisms have been mapped to design and implementation mechanisms in SOLA.

<table>
<thead>
<tr>
<th>Analysis Mechanism</th>
<th>Design Mechanism</th>
<th>Implementation Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>Relational Database</td>
<td>PostgreSQL</td>
</tr>
<tr>
<td></td>
<td>Object Relational Mapping (ORM)</td>
<td>Java Persistence Architecture API (JPA) and Hibernate Java</td>
</tr>
<tr>
<td>Analysis Mechanism</td>
<td>Design Mechanism</td>
<td>Implementation Mechanism</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Persistence Framework</td>
<td>Data Replication</td>
<td>Slony-I</td>
</tr>
<tr>
<td>Communication</td>
<td>Inter Process Communication (IPC)</td>
<td>Java API for XML Web Services (JAX-WS), Glassfish Metro</td>
</tr>
<tr>
<td>Business Logic Encapsulation</td>
<td>Business Interfaces</td>
<td>Enterprise Java Beans (EJB), Dependency Injection</td>
</tr>
<tr>
<td>Transaction Management</td>
<td>Distributed Transactions</td>
<td>Java Transaction API (JTA)</td>
</tr>
<tr>
<td>Security</td>
<td>Authentication</td>
<td>Username Authentication with Symmetric Key and Trusted Subsystem Model, WS-**, Glassfish, Metro</td>
</tr>
<tr>
<td></td>
<td>Authorisation</td>
<td>Role Based Access Control (RBAC), Glassfish Roles and Groups, Declarative Authorisation</td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td>WS-* protocols, Symmetric Data Encryption</td>
</tr>
<tr>
<td>Auditing</td>
<td></td>
<td>DB Triggers, Log4J</td>
</tr>
<tr>
<td>Workflow</td>
<td>State Transition Modelling</td>
<td>Custom state/status attribute on key entities</td>
</tr>
<tr>
<td>Error Management</td>
<td>Structured Exception Handling and Logging</td>
<td>Log4J logs, Web Services Fault Contract</td>
</tr>
<tr>
<td>Document Generation &amp; Reporting</td>
<td>Report Generator/Tool</td>
<td>JasperReports</td>
</tr>
<tr>
<td>User Interface</td>
<td>Desktop Client</td>
<td>Java Swing, MVC pattern, beans binding</td>
</tr>
<tr>
<td>Electronic Data Interchange</td>
<td>Flat file and XML</td>
<td>Comma Separated (CSV), LandXML</td>
</tr>
<tr>
<td>Notifications</td>
<td>Email</td>
<td>JavaMail API, Simple Mail</td>
</tr>
<tr>
<td>Analysis Mechanism</td>
<td>Design Mechanism</td>
<td>Implementation Mechanism</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Document Format</td>
<td>Transfer Protocol (SMTP)</td>
</tr>
<tr>
<td>Printing</td>
<td>Client Side Printing</td>
<td>Portable Document Format (PDF)</td>
</tr>
<tr>
<td>Geographical Information System (GIS)</td>
<td>Spatial Data Persistence</td>
<td>PostGIS extension for PostgreSQL</td>
</tr>
<tr>
<td>GIS Library</td>
<td>Geotools</td>
<td></td>
</tr>
<tr>
<td>Desktop GIS</td>
<td>Custom Swing component</td>
<td></td>
</tr>
<tr>
<td>Image Manipulation</td>
<td>Image Manipulation Library</td>
<td>Sanselan, PDF-Renderer</td>
</tr>
<tr>
<td>Localization</td>
<td>Screen Labels</td>
<td>Java Locale, Swing and Java Resource Bundles</td>
</tr>
<tr>
<td>User Messages</td>
<td>Java Locale and centralized message processing utility using Java Resource Bundles</td>
<td></td>
</tr>
<tr>
<td>Reference Data</td>
<td>Database function to obtain translations for display values</td>
<td></td>
</tr>
<tr>
<td>Currency</td>
<td>Java Locale, currency / money utility</td>
<td></td>
</tr>
<tr>
<td>Calendar</td>
<td>TBC</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2:** Mapping the analysis mechanisms to design and implementation mechanisms (Source: SOLA Software Architecture Document, 2011)

The system architecture of SOLA comprises 3 tiers consisting of a presentation, a services and a data layer. Figure 8 shows an overview of the layers interfaced to external systems and to common functionalities, such as messaging, help etc..
For a detailed description of the layers, their content and the software solutions used in the implementation the reader is referred to the SOLA Software Architecture Document.

The first release of SOLA was presented at the World Bank’s 2012 Conference on Land and Poverty.

SOLA was tested in three pilot implementations in Ghana, Nepal and Samoa. Each pilot implementation was carried out by a local team of software developers in the concerned country. Prior to customization the team was trained by a SOLA mentor. The training allowed the local teams to implement the customization and provide support and maintenance at a local level.

In addition to the customizations in Ghana, Nepal and Samoa, SOLA has also been customized for Lesotho, Tonga and Nigeria. There is now the beginnings of a community to maintain and work with this open source developed software, which is essential for its continuity.

SOLA is one system amongst many possible open source solutions for land administration and cadastre requirements. Land administration systems based on open source have been implemented by various
organisations. For example open source software is being used by Ireland’s Property Registration Authority (PRA), the state organisation responsible for all land and property registration transactions. Most of the software solutions used by the PRA are based on proprietary software, with the exception of the web application “landdirect” which is based on OSS. Landdirect, which is open to external customers, utilises GeoServer to publish spatial data retrieved from the integrated title registration information system database. The application displays maps which show land registry boundaries over a background map or orthophoto supplied by the Ordnance Survey of Ireland. It also allows the user to display and retrieve property details. Figure 9 shows a screenshot taken of the landdirect web site.

Figure 9: Screenshot of landdirect (Source: www.landirect.ie; Property Registration Agency of Ireland, 2015)

In Germany most regional spatial data infrastructure solutions have been implemented on the basis of open source software. The Bavarian Administration for Surveying and Geoinformation uses self-developed tools based on open source software to capture / store data and deliver data via web services (Seifert, 2010, pp. 59-63). The administration decided to implement its digital cadastral and land administration systems using OSS because any available proprietary software would have needed extensive customization to meet their requirements. Figure 10 shows the elements of the Bavarian
surveying and geodata management system. The customization of the OSS was undertaken by software developers employed by the Bavarian Administration for Surveying and Geoinformation. These developers also implemented applications and tools to manage and administer the cadastre and geospatial data. The Bavarian cadastral data was integrated into the national spatial data infrastructure through several web services which are also based on open-source software. The positive experience of the Bavarian land information administration with OSS led to the Bavarian government requesting its public administration to implement, as far as possible, OSS for all information technology requirements. The benefits of open source as viewed by the government are multiple - reduced software licensing and software maintenance costs, independence from software vendors, and software updates and bug fixes quickly and efficiently implemented.

Figure 10: Elements of the Bavarian surveying and geodata management system (Source Seifert, FLOSS in Cadastre and Land Registration, 2010)

A further German example is that of the city of Bonn. Many of the city’s administrative departments, including the cadastre and land administration services, have implemented technology solutions using OSS. Figure 11 shows the current system infrastructure technology of the Bonn geodata management
system and the OSS used to develop it. More than two thirds of the city’s 30 administration authorities input and maintain data on a daily basis. The entire system is essentially always current.

Figure 11. Bonn geodata management system: system infrastructure (Source Bonn City Administration 2014)

Conclusion
Land and cadastre authorities or agencies who decide to use open source software to develop their cadastre and land registration systems (LAS) have access to all necessary OSS components to build their systems. The developer of such a system is not burdened with licence fees or usage restrictions. The concept and design of such a system can be performed based on international standards and its development follows agile methods in order to build a reliable and fully functioning cadastre and land administration system.

LAS software development must distinguish between country independent and country dependent application requirements. The reference architecture proposed here is based on a generic implementation
of LADM which covers the core functionalities of any cadastre and land registration systems. Country specific applications can be developed using any available API.

Due to the nature of the open source development model a core LAS can be developed collaboratively by any interested community and improvements can be shared and used by all. The initial tailoring of an LAS system is one part of the overall cost and only a minor part especially as these systems will be operational for many years. The collaborative approach significantly reduces the total cost of ownership of an LAS.

The use of open source software provides a good opportunity for developing countries to own a cost efficient land administration software solution which can ensure sustainable development

References


**Figures**

Figure 1: Software solutions utilized by the European Commission in the year 2000 (source Damas, Directorate-General for Informatics (DIGIT) of the European Commission, 2014) .......................................................... 6

Figure 2: Open source software solutions utilized by the European Commission in the year 2014 (source Damas, Directorate-General for Informatics (DIGIT) of the European Commission, 2014) ......................... 7

Figure 3: Overview of the conceptual schema of the ISO LADM (Source: International Organization for Standardization, 2012) ...................................................................................................................... 10

Figure 4: Basic classes of ISO LADM (Source: International Organization for Standardization, 2012) ................................................................................................................................. 10

Figure 5: Layered model of the technical implementation of an LAS (Source: Hansa Luftbild, 2015) ....... 12

Figure 6: Diagram of a high level overview of the proposed system and its components (Source: Hansa Luftbild, 2015). ................................................................................................................ 16

Figure 7 Architecture of SOLA (Source: Alexander Lokhman et al, 2012) ........................................ 20

Figure 8: SOLA package overview (Source: SOLA Software Architecture Document, 2011) ............... 23

Figure 9: Screenshot of landdirect (Source: www.landdirect.ie; Property Registration Agency of Ireland, 2015) ................................................................................................................................. 24

Figure 10: Elements of the Bavarian surveying and geodata management system (Source Seifert, FLOSS in Cadastre and Land Registration, 2010) .................................................................................... 25

Figure 11. Bonn geodata management system: system infrastructure (Source Bonn City Administration 2014) ................................................................................................................................. 26

**Tables**

Table 1: Important factors that influence software architecture (Source: Hansa Luftbild, 2015)............... 15
Table 2: Mapping the analysis mechanisms to design and implementation mechanisms (Source: SOLA Software Architecture Document, 2011)