

Using FLOSS towards Building Environmental Information Systems

Dr. Eng. Kostas Karatzas and Asteris Masouras

Aristotle University, Department of Mechanical Engineering, 54124 Thessaloniki, Greece

Abstract: Public access to environmental information is the basis for a higher degree of involvement of citizens and stakeholders in environmental decision-making [Haklay 2003][EU ISPO 1999]. Environmental Information Systems play a key role in contemporary urban environmental management strategies, and are a prerequisite for the proper, timely information of the public [Kampinnen 2001]; yet the fuzzy nature of environmental information [Denzer 2002] requires for systems that can make optimum use of informatics and telecommunications infrastructures to address environmental management needs, while remaining open-ended, easy to use and inexpensive to implement and operate. In this paper, we will attempt to present the characteristics of FLOSS software that render it appropriate for use in developing Environmental Information Systems, accompanied by real world project examples.

Keywords: Environmental Informatics, Environmental Information Systems, FLOSS, Free Software, Libre Software, Open Source

1. INTRODUCTION

Free - Libre - Open Source Software (FLOSS, Infonomics, 2002) is a new software development paradigm that emerged in the last decade and relies directly on the volunteer efforts of geographically dispersed developers of varying professional affiliations and proficiencies.

In direct contrast with previously established business practices [Raymond, 2000], this software development paradigm is fuelled by full disclosure of the source code, volunteer effort and a number of “freedoms” granted to the software user regarding his ability to interact with the software and propagate its use.

By promoting code reuse and the adaptation of freely available best practices, FLOSS development practices minimize redundancy and concentrate investment on innovation [Von Hippel 2003]. The support FLOSS projects receive from the user-developer community serves to provide guidance, reduce maintenance costs and enhance software sustainability, while the service-oriented model of FLOSS allows for a broad range of contractors to provide support, and helps in minimizing the Total Cost of Ownership.

It is these characteristics FLOSS, as we will demonstrate in this paper, that render it flexible, economical and reusable, and thus appropriate for

use in building publicly funded ICT projects [Infonomics, 2002], especially those aiming at the dissemination of information to citizens, such as online environmental portals.

2. USING FLOSS SOFTWARE RESOURCES

Historically, although the software model itself could be said to derive from UNIX, the FLOSS development community and underlying ideological movement is a little more than a decade old: it was officially set in motion with the first version of the GNU General Purpose License (1989) and Linus Torvalds decision to release the Linux kernel to the public (1991). FLOSS represents a software development paradigm, and as such, it is fairly new, compared to its precursors whose roots go back to the '50s and '70s.

The FLOSS development community consists of individuals or groups of individuals who contribute to a particular FLOSS product or technology: as a consequence of the previous statement, this also includes the users of the software. Although referencing various forms of voluntary affiliation around FLOSS projects, the community is the driving force of FLOSS software development. It constitutes a Community of Practice (CoP) [Kimble 2001], and its motivations and processes have been recorded elsewhere in

detail [Ghosh], [Shah], [Lerner 2001]. CoP's are described as "intrinsic conditions for the existence of knowledge", [Lave 1991] attested to by the fact that the FLOSS community provides fertile ground for user-consumer involvement in online joint innovation [Hemetsberger 2003]. The FLOSS process refers to the approach for developing and maintaining FLOSS products and technologies, including software, computers, devices, technical formats, and computer languages.

The definition of Free Software recognizes some fundamental freedoms as imparted by the author (<http://www.gnu.org/philosophy/free-sw.html>) to the user, inside a license agreement:

- The freedom to study how the program works, and the freedom to adapt the code according specific needs
- The freedom to improve the program (enlarge, add functions);
- The freedom to run the program, for any purpose and on any number of machines;
- The freedom to redistribute copies to other users.

The Open Source definition (<http://www.opensource.org/docs/definition.php>) further extended these principles and focused on the development process rather than the political ideology underlying the Free Software movement. The terms Open Source and Free Software refer to software developed and distributed on the above principles, with terms such as Libre software [EWGLS, 2001], or the FLOSS aggregate used to describe them together [Infonomics, 2002]. Although these terms are not fully interchangeable, this paper focuses on the software development process common to both movements.

Unrestricted access to the software source code is a precondition for most of these freedoms, and it is implied that the usefulness and potential for reuse of such software is dependent on the continual revision and adaptation of its source code. In proprietary and closed development environments, the frequency of revisions is dominated by the sales cycle but can also be stilled by managerial decree. In FLOSS, the "life expectancy" of software developed in is a direct outcome of its popularity with developers, who will choose to devote time to improve functionality, and users, who will provide constant feedback to developers on needed improvements and fixes.

The use of FLOSS software towards building environmental information systems hinges on three points [IDA, 2002] providing benefits to users, developers and operators of the software: *economy*, *quality* and *philosophy*.

Economy

Reusing and adapting freely available best practice software, instead of resorting to monolithic proprietary solutions or developing everything from scratch leads to minimizing redundancy in development efforts and by extension, in concentrating investment on innovation. Relying on the community to spark developer interest in the software and provide user feedback reduces maintenance costs and prolongs its' useful life cycle. A corollary of this is that the functionality and maintainability of the software is not impaired by artificial limitations (i.e. not intrinsic to the software itself), such as expiring licenses and financial plights affecting a single developing entity.

The Total Cost of Ownershipⁱ (TCO) of solutions based on FLOSS from a contractor point of view is alleviated [EWGLS, 2001, The Mitre Corp., 2001], since consulting fees are fully useful expenditures, in contrast with licensing fees which mostly serve as instruments of amortization for developing companies. Since Environmental Information Systems development is largely supported by public funding, such amortization should not burden beneficiaries of their services. For the service-oriented model of FLOSS, it should be noted that costs of support and maintenance can be contracted out to a range of suppliers, as per the competitive nature of the market ensured by source code disclosure [Lerner and Tirole, 2001].

Quality

The main objective in software engineering is not necessarily to spend less but rather to obtain a higher quality for the same amount of money, and aim to enforce the best possible safeguards for quality and safety in the product. Avoiding to "reinvent the wheel" by using funds to develop new applications rather than re-inventing already developed parts, speeds up technological innovation -as is also the case with the increased cooperation and full source code disclosure and availability required by FLOSS tenets. Finally, as has been repeatedly demonstrated in recent years [Perens, 2001, Schneier 2001]), software security concerns are better addressed through a continuous process of issue disclosure and user-developer cooperation in order to overcome them.

Philosophy

FLOSS presents the potential for a Social Return of Investment on public funding, by virtue of constituting a Global Public Goodⁱⁱ [UNDP, 2002], and by its' potential to produce non-monetizable benefits for society, in the form a body of code

that can be utilized in building sustainable informatics infrastructures for the public.

Reliance on proprietary software for science results in vendor “lock-in” as regards to data formats, making it difficult to pursue common protocols for data interchange and storage, for instance, as it is required by modern systems dealing with the problems of environmental data heterogeneity [Visser et. al, 2001]. In contrast, FLOSS developers and proponents promote the use of open scientific standards, through their use in applications, as a means of consolidating researcher efforts, minimizing the cost and dependencies of technical innovation. In addition, the FLOSS software movement serves the further collaboration between public bodies, professional communities and the private sector in the interests of creating a flexible and lasting service environment for the publicⁱⁱⁱ. The free dissemination of technological advances (both in terms of cost and material availability) relating to informatics services, although not a panacea, can be seen to eventually help eclipse the digital divide [Schauer, 2003], by allowing poorer countries to “catch up”.

3. ENVIRONMENTAL INFORMATION SYSTEMS ASPECTS

Environmental Information Systems are informatics systems concerned with the management of data about the status of the environment and related scientific, regulatory, legal, managerial or other information, and are used by authorities, policy and decision makers and or experts for environmental monitoring, management planning and coordination, environmental impact assessment, urban planning and decision support. etc. Due to the complexity of the decision processes involved, disparate data from a variety of sources must be combined. This “holistic nature” of environmental information systems leads to heterogeneity problems regarding the syntax, structure, and semantics of environmental data [Visser 2001]. Overcoming them requires, among others, the adoption of common protocols for data exchange and storage, and making use of metadata to facilitate interoperability between subsystems.

FLOSS promotes the use of open data standards as a means of consolidating researcher efforts and increasing technical interoperability. Thus it can be demonstrated that the right of public access to environmental information, as has been defined in contemporary legislation [EU/EC 2003a][EU/EC 2003a], is better served by utilizing open, flexible and low cost dissemination platforms that make

use of software developed by the community. In the following chapter, examples of FLOSS applications are presented, all related to air quality management systems and all addressing problems that converge into the need of openness, flexibility, adaptability, resource optimum, environmental management solutions.

4. PROJECT EXAMPLES

In the following, we demonstrate the utilization of FLOSS resources towards building public Environmental Information Systems [Haklay 2003], on the basis of EU supported projects. In the core of our approach, we realise Environmental Information being a public good and in parallel the raw material for the compilation of electronic information services. FLOSS may then be considered as a “public good” in the sense of publicly available software infrastructure and functionality potential that may be used for the development of electronic environmental information services to support personal well being in accordance with environmental awareness raising, thus framing a “healthy” sustainable development paradigm. To this end, the human-centred approach of Environmental Informatics applications may be served.

4.1 The APNEE/APNEE-TU projects:

The APNEE project (<http://www.apnee.org>) contributed to the European research on public information systems and services, by developing citizen-centered dynamic information services aimed at providing intelligence about the ambient environment. These services advise the citizen about the air quality in terms of air quality indexes and offer guidelines for behavioural change. Awareness services are based upon an array of information channels to reach the citizen. APNEE further utilises various intuitive presentation formats to convey information. The configuration of such ambient technologies and the selection specific information channels has been evaluated in field trials in different European regions. APNEE-TU further investigated with success the feasibility and adaptability of the APNEE approach in relation to new technologies, extended and updated content, and new application sites

It was apparent from the beginning of the APNEE project that a flexible, modular and cost-effective architecture was needed, to support the environmental information needs of urban agglomerations through easy-to-use and easy-to-access interfaces that would allow a measure of personalization / customization in order to prove attractive to citizens. For this reason, development

of the APNEE regional server was based on FLOSS technologies.

APNEE / APNEE-TU is composed of a set of reference core modules, including the database, the service triggers, the regional server application and basic functionality modules (licensed as Open Source), as well as proprietary extension modules developed by telecommunication partners to provide services based on local ICT infrastructure conditions. Although the core modules are considered to be the heart of the system, yet, they may be “by-passed” or not implemented, in cases where only the electronic services are of interest for installation and operational usage, provided that there is a database and a pull and push scheme provided via alternative software infrastructures. The modules and the interfaces that have been developed by the project consortium, to build the regional server, are the following:

- **APNEE Environmental Database:** the database forms the back-end of all APNEE-TU services and consists of a schema for environmental data series, as well as warnings, medical advises, pollutant information; spatial data for the WebGIS component, and user information and personalization data for subscribers. APNEE-TU provides an object-relational persistence layer to allow cooperation with a variety of FLOSS and proprietary RDBMS systems.
- **APNEE Regional Server:** the centerpiece of the APNEE platform, the regional server provides a web-based anchoring point for APNEE services, configured and localized as per the needs of each installation site; also provides administrative interfaces for a variety of functionalities, such as subscription to the newsletter, and email services. The materialisation of the regional Server for Thessaloniki-Greece is presented in Figure 1.
- **Push services:** these services consist of modules that are executed on when changes in the database occur. What kind of database change that will launch a module, is configured in the *trigger*. Push services consist of sending SMS and email messages to the citizen periodically, or upon user specified conditions, and are mostly used to send out alerts and warnings.
- **Pull services:** these services are used whenever another application requests information from the database. This includes requests made from users via WWW, PDA, or WAP, and requests from automatic processes using the XML-RPC or SOAP interface.

In addition, a variety of technologies was used for the development of electronic environmental information service applications, including:

- **J2ME Applications:** Based on Java 2 Platform, Micro Edition (J2ME) for mobile devices, these applications serve the need for on-time information of remote users. They require to be downloaded to a J2ME-enabled mobile phone and provide static and dynamic information which is updated by using GPRS and http protocols.
- **PDA web application:** This is a “lighter” version of the regional server, with trimmed layout, navigation, images and content to allow for the smaller display size of the average PDA. It allows the browsing of web application via a PDA, by automatic adaptation to the device characteristics to display the same content and give access to the same bundle of information services.

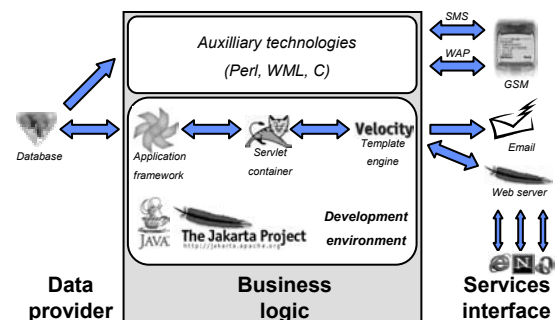


Figure 1. The APNEE-TU services architecture materialisation for Thessaloniki, Greece.

Implementation of the APNEE regional server (ARS) was based on Java servlets, server-based web applications, and a variety of technologies made available by the Apache Foundation (Figure 3). At the inception of APNEE, in early 1999, Java 2 Enterprise Edition (J2EE) had not been finalized, while the first FLOSS J2EE-compatible container was certified in 2002. The ARS, while nominally a J2EE application, does not take full advantage of the J2EE framework, as it was based on Jakarta Turbine, a modular service oriented web application framework that provides the Torque object-relational mapping layer, the Velocity dynamic HTML-based template language, a comprehensive RBAC security system that incorporates groups of users, a templating framework and an intra-application service management layer for pluggable services. The templating framework of Turbine worked very well in our case, allowing separate teams to focus on presentation and logic, in conjunction with the

automatic build and deployment scripting system based on Apache Ant. We found out that the Turbine security model is rich enough for our need, but that it does not map too well to the J2EE security model. The ARS programmatically provided service front-ends to the environmental database through Torque, as well as Web Services interfaces, via XML-RPC and Apache Axis. Finally, Jakarta Tomcat was chosen as the default servlet container for both development and production use.

In recognition of the value of the FLOSS software paradigm towards building informatics systems for the public, and in order to preserve and disseminate development efforts, the APNEE-TU project produced a “reference implementation”, composed of the environmental database, regional server and core modules, which is licensed as Open Source, thus making its embracement and support of use cases by anyone interested much easier.

4.2 APNEE Use Cases:

Citizens might daily traverse several parts of the city, in commuting to and from their places of work, and in attending to personal and family needs (access to local markets and public services etc.).

- Certain population groups being sensitive to air quality levels, as they can affect their health, especially in regard with the respiratory system, leads to specific needs for environmental information.
- The parents of children affected by respiratory problems, allergies and other related health problems would benefit from advance warning, based on scientifically authoritative predictions, on possible increases in air pollution levels, the discomfort index and any other parameter that affects the quality of the urban environment in residential areas, school districts etc.
- People with respiratory problems would be interested in being informed early on sudden increases of air pollution levels, before commuting to their places of work, or visiting friends and relatives in remote parts of the city
- Elderly people and performers of strenuous physical exercise or continuous labor in open spaces would also be interested in timely and authoritative information on the state of the ambient air, and the possible negative health effects of air pollution.

Citizens concerned about these and other issues connected with the urban environment can make use of the environmental information services offered by the APNEE portal, through the multitude of complementary communications

channels supported and made possible by the technological advances of the Information Society.

5. CONCLUSIONS

The need for collaborative environmental information management and dissemination modules that would allow for implementing a homogenized, service-based, user perspective of heterogeneous data and computational resources was the main drive towards modular and open software architectures in projects such as the ones previously mentioned. It was also made apparent that project design and development could benefit from the use of Free/Libre/Open Source Software. This was mainly initiated within the last decade via the usage of Internet based communication infrastructures, leading to the flourishing of platform-independent (eg. Java based) software module implementations and the need for cost-effective and reliable informatics solutions. APNEE/APNEE-TU followed the trends outlined above and provided a flexible, cost-effective working solution for implementing a public environmental information ICT infrastructure, making use of resources developed by the aggregate FLOSS community.

6. ACKNOWLEDGEMENTS

The authors greatly acknowledge the European Commission for supporting research projects APNEE (IST 1999-11517) and APNEE-TU (IST 2001-34154), and their partners herein.

7. REFERENCES

- Böhler, T., K. Karatzas, G. Peinel, Th. Rose and R. San Jose, Providing multi-modal access to environmental data—customizable information services for disseminating urban air quality information in APNEE, *Computers, Environment and Urban Systems*, 26(1), 39-61, 2002
- Chan, T., J. Lee, A Comparative Study of Online User Communities Involvement, Product Innovation and Development, 2004, <http://opensource.mit.edu/papers/chanlee.pdf>
- Denzer, R., Generic Integration in Environmental Information and Decision Support Systems, *ieMSS Proceedings*, 2002, <http://www.iemss.org/iemss2002/proceedings/pdf/volume%20tre/denzer.pdf>
- European Parliament, European Council, Directive 2003/4/EC on public access to environmental information and repealing Council Directive 90/313/EEC, 2003

- European Parliament, European Council, Dir. 2003/35/EC, 2003a
- European Parliament, European Council, eEurope 2002: An Information Society For All Action Plan, 2002.
- EU ISPO, Public Sector Information: A key resource for Europe, COM(98)585final, adopted on 20 January 1999.
- EWGLS-European Working Group on Libre Software, Free Software / Open Source: Information Society Opportunities for Europe, 2001, <http://eu.conecta.it/paper.pdf>
- Ghosh, R. A. Understanding Free Software Developers: Findings from the FLOSS Study <http://opensource.mit.edu/papers/ghosh.pdf>
- Haklay, M., Public Access to Environmental Information: Past, Present and Future, *Computers, Environment and Urban Systems*, 27, 163-180, 2003.
- Hemetsberger, A., When Consumers Produce on the Internet: The Relationship between Cognitive-affective, Socially-based, and Behavioral Involvement of Prosumers, 2004, <http://opensource.mit.edu/papers/hemetsberger1.pdf>
- IDA-European Commission, DG Enterprise, Pooling Open Source Software An IDA Feasibility Study, 2002.
- Int. Institute of Infonomics, Berlecon Research, ProActive Int., Free/Libre and Open Source Software (FLOSS) Survey and Study, 2002, <http://www.infonomics.nl/FLOSS/index.htm>
- Kamppinen, M., P. Malaska, M. Wilenius, Citizenship and ecological modernization in the information society. *Futures* 33, 219–223, 2001.
- Kimble, C, P. Hildreth, P. Wright, Knowledge Management and Business Model Innovation, Chapter 13, 220 – 234, 2001, <http://www.getcited.org/pub/103396967>
- Lave J. and E. Wenger, Situated learning. Legitimate peripheral participation Cambridge University Press, 1991, <http://citeseer.ist.psu.edu/context/98865/0>
- Lerner, J. and J. Tirole, The Simple Economics of Open Source, 2002, <http://ideas.repec.org/p/nbr/nberwo/7600.html>
- Lerner, J., and J. Tirole, The open source movement: Key research questions, *European Economic Review* 45, 819-826, 2001.
- Mitre Corporation. A Business Case Study of Open Source Software, 2002, http://www.mitre.org/work/tech_papers/tech_papers_01/kenwood_software/kenwood_software.pdf
- Neus, A., Managing Information Quality in Virtual Communities of Practice, *The 6th International Conference on Information Quality at MIT*, 2001, <http://citeseer.ist.psu.edu/567684.html>
- Nichols, D.M., K. Thomson, S.A. Yeates, Usability and open-source software development, *Proceedings of the Symposium on Computer Human Interaction*, (eds.) Kemp, E., Phillips, C., Kinshuk & Haynes, J., 6 July 2001, Palmerston North, New Zealand. ACM SIGCHI New Zealand. 49-54. ISBN: 0-473-07559-8.
- Perens, B., “Why Security through Obscurity Won't Work”, 2001, <http://slashdot.org/features/980720/0819202.shtml>
- Raymond, E. S., The Cathedral and the Bazaar, 2000, <http://citeseer.ist.psu.edu/context/66413/0>
- Schauer, Th., The Sustainable Information Society, Visions and Risks, 2003, <http://www.global-societydialogue.org/saskia.pdf>
- Shah, S., Understanding the Nature of Participation & Coordination in Open and Gated Source Software Development Communities
- Schneier, B., “Full Disclosure”, Crypto-Gram Newsletter 111, 2001, <http://www.counterpane.com/crypto-gram-0111.html>
- The Apache Foundation, Apache Web Services Project, <http://ws.apache.org>
- The Jakarta Project – Open Source Java-based tools. <http://jakarta.apache.org/>
- United Nations Development Programme, Global Public Goods Economic Briefing Paper no.3 on Sustaining Our Global Public Goods, Earth Summit 2002, <http://www.earthsummit2002.org/es/issues/GP/G/gpg.htm>
- Visser U., H. Stuckenschmidt, H. Wache, Th. Vögele, Using Environmental Information Efficiently: Sharing Data and Knowledge from Heterogeneous Sources, 2001, <http://citeseer.nj.nec.com/309536.html>
- Von Hippel, E. and G. Von Krogh, Open source software and the private-collective innovation model: issues for organization science. *Organization Science* 14 (2), 209–233, 2003.

ⁱ “The effective, combined cost of acquisition and deployment of an information technology throughout all its perceived useful life” [EWGLS 2001]

ⁱⁱ Earth Summit 2002, Economic Briefing Paper no.3 on Sustaining Our Global Public Goods denotes “free access to basic and essential software” as a “key policy option”.

ⁱⁱⁱ The 2002 and 2005 Action Plans of the European Commission’s eEurope initiative promoted FLOSS and open standards for use in the public sector and e-government.