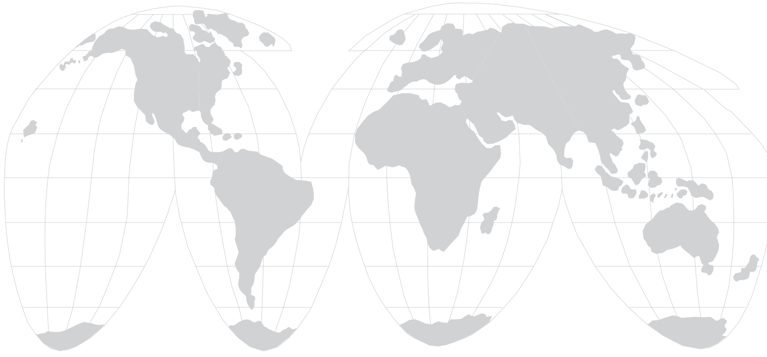

How to design an open(source) e-learning platform. The ADA experience

Stefano Penge, Maurizio Mazzoneschi
and Morena Terraschi

Lynx s.r.l.

steve@lynxlab.com

Abstract

When speaking about «open» e-learning platforms, we generally mean that code should be readable and modifiable.
While the advantages for programmers of adopting this developing method are well known, in this article, we look at it from a different perspective, assuming that the meaning of «open» may be extended to every user (open to programmers, but also open to teachers and to students) and to different levels (code level, document level, structure level, and interface level).
We try to investigate the full meaning of the expression «open platform» from a pedagogical, technological and ethical point of view, by means of a description of the architecture of an opensource platform, ADA.
We discuss modularity, accessibility, knowledge construction and evaluation in an open perspective and, finally, we outline a definition of open platforms as *learning* platforms.

1. Introduction: a broader meaning for «open»

When speaking about «open» software, we generally mean that code should be readable and modifiable. Certainly, opensource has been a revolution and has changed the scenario of technology application in education, especially as regards the development and administration of web applications like e-learning platforms.

While all this is common knowledge, when we look closer at the «open» concept, some questions arise.

Many opensource platforms[1] can often be as performant and robust as proprietary ones. However, we may ask ourselves if our effort to build opensource platforms is worthwhile. How many users will benefit by its openness? And what type of benefit/s will they reap? Is the simple fact that our software will be openly developed a real improvement from all points of view?

In this article we intend to demonstrate that there is, indeed, a close relationship between the «open» concept and the digital learning environment, assuming that the meaning of «open» is extended beyond the application code.

Let us discuss the first question: how many users are directly affected by opensource in e-learning?

The idea behind opensource («if you don't like the way a software behaves, just fix it»)[2] has proven very powerful and seems indeed extremely attractive to all of us with a *do-it-yourself* attitude.

While the possibility to modify source code is, in principle, positively welcomed by all users, the size and complexity of real life computer applications goes far beyond the competences of most of us. It is a fact that the competences required to modify an opensource software are not trivial; this simply limits the profile of users genuinely concerned by the «open» question only to professional programmers. The end-user of a software normally cannot — or simply doesn't care to — debug or modify it.

This is especially true in the education field. Teachers — as well as school or university administrators — like opensource platforms because they are free, not because they are open. Perhaps they are aware that the openness of source code has secondary advantages. They believe that the overall technical quality of opensource platforms is often high, because a lot of users can read the code and fix bugs. From an ethical point of view, open[3] software can be an interesting model for the relationship between work and market to be discussed in a history course.

[1] http://www.unesco.org/webworld/portal_freesoftware/Software/Courseware_Tools.
[2] http://www.opensource.org/advocacy/.
[3] We are not so rigid here in distinguishing between open software and free software because teachers often do not.

As another collateral effect, it encourages schools to redirect economic resources normally spent in acquiring software licenses from international software houses toward young local programmers — perhaps toward schools' alumni.

So far, however, openness does not directly affect most e-learning users, such as teachers and students.

To be really meaningful to them, «open» should mean that *all* users (not only programmers, or IT administrators) can modify and re-arrange a digital learning environment (or a small part of it) to satisfy their needs. We should extend the notion of «open» to include every user: open to programmers, but also open to teachers and to students.

We should borrow the definition of «open» from other contexts, like the «open university». An open platform should be flexible, adaptive, centered on user's needs more than on technology.

If we intend to follow this direction, we cannot limit discussion to source code of software, but rather we should speak of «open» at different levels: code level (open *source*) but also document level (open *content*), structure level (open *architecture*) and accessibility (open *interface).* We should also start to consider open *evaluation.*

The second — and perhaps main — question is: why is it so important to design «open» (as in the definition outlined above) e-learning platforms? This leads us to discuss the pedagogical theory that underlies platform design.

Constructivism — probably the most widely accepted theory of learning today — says learning is not just acquiring knowledge. From Ausubel to Jonassen, learning has been recognized as (a) a contextual activity that modifies (b) existing learner's concepts and (c) the way concepts are organized in her/his mind.

Environment — the context — is a fundamental element of the learning process; it has an influence upon *what* we are learning and also upon *how* we learn. And when we have finished restructuring our knowledge, we start to rearrange the context itself.

We can think of learning as the process by which a subject — human or not — takes control over an environment. At the beginning s/he is relatively passive, and only reacts to stimulations that come from the environment. Progressively, s/he becomes more and more active, understands facts and makes some generalizations, until s/he is able to modify environment rules. Control over the situation changes hands from the environment to learner.

Learning is a form of inquiry aimed to clarify non-determined situations.[4] While learning, we shift the border line between what we know and what remains

[4] This is a free re-formulation of Dewey's transactional approach to knowing which explicitly includes environment; an approach that we find very suitable to go beyond constructivist theory of learning. See Dewey and Bentley, 1949.

unknown. We constantly restructure our mental representation of the environment (as we do, for example, after a first glance at a new software interface); but the more we know about environment rules and feel comfortable with them, the more we try to modify the environment itself. While adapting ourselves, we also adapt the environment to our needs. Our success in doing this is the main indicator of our new competences.

In real life this can be a long and dangerous process, so we created *protected* educational environments, like schools, that are expressly designed to progressively let the learner take control.[5]

This is why digital learning environments must be designed in an «open» perspective. To truly allow for learning to take place, a platform (like a digital learning environment) has to be designed explicitly to be modified by users. If we really want students (but also tutors and authors) to learn on-line, we should create dynamic, always modifiable — at various levels — digital environments.

The remainder of this article is dedicated to the explanation of the full meaning of the expression «open platform», intended as a more general perspective that brings together pedagogical, technological and ethical issues, by means of the description of the architecture of an opensource platform, ADA.

2. Why

The ADA[6] project was started in 2000 by a group of researchers from a small software house with mixed professional backgrounds, ranging from computer science to pedagogy.

At that time, some[7] opensource e-learning platforms already existed, but the basic idea behind the new project was to build an «open platform» for e-learning: not only an open-source platform, but an open one in a broader sense of the word.

We planned from the beginning to release the code under the GPL license and to make it possible to download the software for free,[8] to build a community of users that would participate and extend the platform. But we were also motivated by non-technical reasons, as we will briefly show below.

[5] The concept of «fading» in cognitive apprenticeship theory is perhaps appropriate to describe the perspective assumed here. See Collins et al., 1989.

[6] ADA is an acronym for «Digital Learning Environment» (Ambiente Digitale per l'Apprendimento). Currently (September 2005) we are going to release version 1.8, with some relevant upgrades. See official web site http://ada.lynxlab.com for details.

[7] Relatively few: Claroline was started in 2000, and ILIAS was registered on Sourceforge in 2001, to cite only two major European opensource platforms.

[8] ADA is also registered in Sourceforge, the most important web repository of opensource software. See http://ada.sourceforge.net.

The first reason was the context in which ADA was to be used. The platform would be used mostly in non-formal educational contexts: primary and secondary schools, volunteer associations, professional communities.[9] Therefore, it had to be designed to fulfill the specific needs of this kind of subjects, more than to reach state-of-the-art in technology.

Moreover, typical e-learning projects in these contexts are very different from those carried out in Universities or in large enterprises.

In this field, the more complex e-learning projects are often faced with scant resources, both human and technological. Band width, processor speed and disk space are limited resources, on both the server and on the client side. More often than not, it is impossible to know in advance which operating system, which browser, and which Internet connection will be available to users.

Platform technical requirements will by necessity be very light. In contrast, on the other hand, everything must be done to enable users (from administrators to teachers) to fine tune the platform to better suit their own needs. Fundamental requirements in these contexts are open architecture, and a mechanism to personalize the platform easily.[10]

This approach calls for a trade off: while designing ADA, every time we had to choose between advanced functions and more largely supported function, we chose the latter.

On the other hand, the educational projects in this area are often strongly committed to major pedagogical principles such as socio-constructivism. It is well known that, in adult education, e-learning cannot be simply thought of as knowledge transmission. However, in the formal education context this widely accepted principle has often to face the need to teach specific and well-defined abilities. On the contrary, in these non-formal contexts, co-construction of knowledge is the rule, and the growth of the entire group knowledge — even though somehow ambiguously defined — is considered more important than the acquisition of skills by single students.

The typical model in an e-learning course of this kind is peer-to-peer education, with no clear distinction between teacher and student, where every participant is in turn teacher and student.[11]

With this horizontal interaction model, it is not rare that e-learning projects are simply supported by collaborative web environments, allowing the exchange of messages in a synchronous and/or in an asynchronous way. We started to design

[9] We started to test ADA within a teachers web community, named Altrascuola. Many of the ideas illustrated in this article are the outcome from discussion of designers with authors, tutors and students. In this sense, ADA has been (co)built by its users. See http://corsi.altrascuola.it.

[10] See below, 2.1.

[11] We can think of this context as a community of practice. See Wenger, 1998.

ADA bearing in mind that an e-learning platform cannot be just another name for a suite of different communication tools (chat, forum, etc). The possibility to build knowledge together depends on the availability of tools that enable all participants, at different levels, to share their knowledge, to structure it in conjunction with each other's one, to personalize it and to reuse it again in different contexts.[12] If knowledge added by teachers («theory») should not be radically different in type from knowledge added by students («experience»),[13] communication tools must really be integrated with knowledge authoring tools.

Here again, we extend the meaning of «open», this time from the technology area to the content area. An open platform should be capable of receiving all kinds of learning content, from all users, and of organizing them in flexible structures that can be dynamically searched, modified and exported.

The problem of copyright, that immediately arises, brings us to the topic of «open content». «Open» does not necessarily stand for «free». Different kinds of license are currently available (FDL, Creative Commons, etc) in contrast to standard copyright management. While this is not the place to go further into this point, it is just worth mentioning that every piece of content in ADA is marked up as «copyright type» property, to allow any licensing policy.[14] In principle — and this is one of the directions for future development in ADA — even links, bookmarks and routes can be licensed in one manner or in another. Not only code and content, but also structure can be open or proprietary. This will no doubt be the next battle field.

This «open» approach would be embedded in all aspects of the ADA platform. Let us look briefly at how this has been done from a technical point of view.

3. How

Some general choices in ADA design can probably be easily foreseen by expert readers at this point.

First of all, from a user point of view, ADA should not be committed to a particular operating system, nor would it require a specific browser or an additional plug-in. All pages sent to browsers by ADA are compliant with W3C specifications for HTML 4 and to WCAG 2.0 recommendations. Possibly, a textual browser such as Lynx can be used to navigate in ADA.[15]

[12] This can be seen as an application of the Nonaka and Takeuchi SECI model of knowledge construction in organization. See Nonaka, 1995.

[13] See below, 3.3.

[14] Normally course authors are not requested to specify under which license they intend to release their content.

[15] This is true for ADA interface, not necessarily for course contents. Authors may decide to embed Flash movies or Java application that require students to download some additional software.

With regard to the server, ADA can be run on every SQL-based DBMS (from MySQL to Postgres, from MSSQL to Oracle) and only requires a web server with a PHP 4.* interpreter.

As for interoperability, we don't want to tie ADA to any specific standard. Technical standards are very important in the enterprise world, where they can help to reduce costs when migrating from one system to another; much less in the educational field, especially if they restrict the authors' didactic creativity to some specific model of content unit and of assessment tool.

We thought that a strict compliance with international e-learning standards such as SCORM or AICC would be excessive in ADA, but to protect authors' work in the future and to simplify data exchange we chose simply to use XML (with a public DTD) as a general, open format to interchange for course data between the authoring module and online platform. This choice allows every user to extract meta-information (from structure to contents) from an ADA course without being tied to a specific software and operating system, and possibly to reuse it within another platform.

Moreover, all texts are internally coded in HTML and authors are asked not to use proprietary file formats (such as PDF, PPT) if not absolutely necessary. This restriction is set not only to improve accessibility, but also to allow full-text search and analysis: ADA offers all users a «dynamical» lexicon, that is an index of words used by authors and students ranked by frequency. But it also allows users to export learning contents by means of a RDF-like mechanism: from outside ADA, another e-learning platform (or web service) can request for a node content and receive just the XML version of it. This i*nterface independence* — more than WCAG compliance — permits disabled students to obtain content in a format more suitable than HTML to be read from a text-to-speech synthesizer.

Looking more closely at the ADA structure, we can now illustrate four main topics: modularity, accessibility, knowledge construction and evaluation.

3.1 Modularity

As seen above, open architecture is a major requirement in standard educational fields. Like many other opensource e-learning platforms, ADA is built with a modular approach. Moreover, the coding style is an object-oriented one; every major function is implemented as a class method. A simplified version of class hierarchy shows like this:

```
Users
   Author
   Tutor
   Student
   Administrator
```

```
Content Objects
   Node
      (Terminal) node
      Group
      Note

   Link
   Bookmark
   Exercise
   Multimedia resource

Interfaces
   XML
   HTML
   HTML_elements
      Table
      List
      Form
```

The database is connected and queried within a single low-level module (AMA, the ADA Middle API) which in turn uses a standard PHP package named PEAR; no SQL query is needed in other parts of the application. This is done to maximize portability and to ensure ease of code maintenance. While this obliged ADA programmers to use only a subset of SQL instructions, installing ADA on a different DBMS simply requires changing the configuration file.

All contents are sent to browsers via a template engine and (almost) no HTML code is directly embedded in the application script, except for that required to format dynamic data (course index, etc).

For this type of data, all HTML elements (tables, lists, forms) are built by means of respective object methods. Once again, code maintenance is much simpler.

In addition, a user-module loading mechanism has been developed which allows programmers to easily write a new module inheriting methods from AMA classes, exploiting template engine/s, and so on.

3.2 Accessibility and usability

As previously stated, ADA interface is completely separate from code: it resides in several HTML *templates* with their associated CSS. These templates are dynamically loaded from ADA depending on user status and on request. For example, the interface for viewing a course element will obviously be different whether requested by a student or by its author; so two slightly different templates have been designed for the same module, but only one of them is loaded at runtime.

This mechanism can be controlled by users. Every module has its own interface, but for a single module, several different «styles» of interface may exist. Style, in this context, means something rather general: fonts, dimension and color, but also content disposition, icons, etc.

Style can hide some irrelevant information, or stress other, in association with the user's cognitive abilities and styles, or, simply, his/her preferences.

Templates are arranged in a hierarchical order, from general to specific, and are the responsibility of the different type of user:

Level	*User*
Platform (all courses)	Administrator
Course (all classes)	Author
Class (all nodes)	Tutor
Node	Student

More specific levels override general levels. When a conflict arises, the specific needs of a student are considered to be more important than those of the stylistic choice of an author for her/his course.

Thus, while an administrator can choose a general style which is well-adapted to the entire web site, an author may prefer a style which is better suited to his course domain. A tutor can simplify the interface so as to meet the navigation abilities of his class, or a student can decide to hide all unnecessary icons and buttons as long as s/he does not need them.

3.3 Knowledge construction

The course author is not the only user who owns the knowledge. An adult student can rarely be regarded as a *tabula rasa* with no knowledge of the course matter. S/he surely has a partial, perhaps incorrect, knowledge; but probably has a lot of experience in a similar field which can be recalled and used by means of analogy.

In ADA — as in every socio-constructivist labeled platform — an e-learning course is viewed as a process by which a (virtual) group can share, structure and personalize knowledge; the platform is the (virtual) place where this process can take place. From this perspective, authors are simply those users who start the process by selecting (or producing) relevant documents and structuring them with the aim of facilitating learning.

Tutors provide more information by means of platform communication functionality (e.g. message, agenda, chat).

However, from the beginning of the course, all participants (students, as well as tutors and authors) are invited to enrich the course's documents with comments, personal notes, links, original multimedia materials and so on. These bits of knowledge are what distinguish every edition of a single course from all the others.

This is possible because in ADA all learning content is managed by means of two main types of objects: nodes and links. Every piece of content in ADA is a *node*, published either by an author, or by a student. A node has some persistent properties, such as «title», «keywords», «author», «type», «level», «parent» and can be enriched with multimedia content (i.e. photos, sounds, external text documents, Internet resources). Nodes can depend on other nodes, but can also be linked to other nodes in a complex hierarchy which is represented for users as a map.[16]

This approach has some interesting consequences. In ADA, the Forum — where students and tutor can discuss course topics — is not a completely different environment, as in others platforms. When a student reads a course unit, s/he can start a discussion on a theme related to the unit simply by adding a note (such as a «post-it») to that unit. Every other user (students, but also tutors) can reply to the note, or to the original unit. Moreover, they can navigate through notes, search for one of them, see an indexed view of all notes of the course, and so on.

Strictly speaking, the Forum is simply a filtered view of the dynamic structure of course, which highlights notes added by students while hiding the authors' contents.

Two more details must be illustrated at this point. The first concerns the collective construction of knowledge, one of the major themes of current reflections on e-learning.

Some of this new piece of knowledge (notes) can be promoted to full *node status*, thus becoming part of the course in its future edition. This can be accomplished by both the tutor (who invites the author to take the new proposal into account) and author (who may decide to accept or refuse it). While it is presently a very basic mechanism, this can still lead to the co-construction of knowledge — not as simply pasting works from several authors or different fields, but — as a continuous refining process among all the participants within a learning community.

The second detail concerns the personalization of knowledge. At the end of an ADA course, every participant can download her/his own version of the course, filtered by keywords and augmented with all notes, bookmarks, and attached documents the student has added. Put in Nonaka and Takeuchi terms, this could be a step between explicit/public knowledge and tacit/private knowledge: a kind of explicit, though private, knowledge.

[16] Although ADA maps are very close to classical conceptual maps of J.D. Novak, they differ in some aspects. Every node is not just a name for a concept, but *has* content itself. Higher nodes in hierarchy (groups) can be exploded to lower nodes, that in turn can contain other nodes; so we could call them «augmented recursive maps». See Novak, 1985.

3.4 Evaluation and assessment

Evaluation in an e-learning context is a subject yet to be fully explored. While it is surely possible to apply standard off-line assessment tools such as tests, we assume here that a digital learning environment allows another, more specific, approach. Evaluation should be considered as an on-line process itself. If we do not want to limit ourselves to assessing students' initial and final competences, but want to know about learning itself, about the process of taking control of the environment, we need much more data than test results.

Probably every e-learning platform today can track student behavior and show tutors what has been read by students, as well as when and how. But lots more data are available because all didactic interaction among students, and between student and tutor, occurs *within* the platform. Every message, every comment, every function activation is recorded by the platform. This can be both an opportunity and an obstacle. Technically speaking, the problem is the so called «garbage collection»: the process by which non necessary data have to be erased — after a backup — to avoid database overload. From an ethical point of view, the problem is to what extent are we authorized to record and analyze these data without violating student (and tutor) privacy. There are, nevertheless, some advantages. The open approach implies that we save these data in a format suitable for subsequent processing, even with external tools. Once again, XML is probably the less efficient, yet the more open choice; that is, the less engaging one for future needs and techniques to appear. We are currently developing a backup system that prevents privacy violation, while allowing for different analyses on group learning.

By processing these data, we can then answer many questions:

– what navigation style do students use in course?
– how much does a student interact with other students and with tutor, and how?[17]
– which terms and expressions are most frequently used in forums and chats?
– how does this distribution vary during course navigation?

The results of these analyses can give insights when evaluating the (possible) evolution of a student, or — more significantly — of an entire group within a course edition. We may also want to compare different course editions, to discover regularities (for instance: a relationship between the tutor's scaffolding messages and the students' navigation style adjustment). This is not only an issue for the researcher, but also a form of self-evaluation practice for tutors when trying to refine their support for learning groups.

[17] An effective tool to manage these data could be Social Network Analysis. See Mazzoni, 2005.

This kind of analysis is computationally too heavy for a web application, and is better accomplished off-line. However, in ADA a less specific, on-line analysis module is available to tutors and students to monitor the direction of an ongoing learning process in real time. Not only tutors, but even students can read a report about their activity with seven parameters (messages sent and received, added notes and node visits, exercises and score, level). Some of these values are compounded in a single index[18] that can be used to «freeze» the global interactivity of the student. Moreover, every student can compare her/his values with the average for the group. These values can also be exported to a data sheet to be submitted for further analysis, or can be represented in a more concise, graphical form.

4. Conclusion: learning platforms?

Open platforms are *designed* to be modified by users. This simple working hypothesis forced us to make choices during every stage of ADA design, from architecture to interface. At the end of this description, we may, perhaps, assume another point of view.

E-learning platforms are complex objects. Sometimes we still think of them just as simple teaching machines, an algorithm repeating the same operations forever. We represent them as tools to support learning, more than as environments where learning takes place. To try to go beyond this simplification, we shall use a metaphor. A digital learning environment can be seen as an organism; after being designed and built, it starts to «live» autonomously, to evolve and to mutate. It is its «natural» way of behavior: there is an interchange of data with authors — who input their course materials — and with students — which put in performance and navigation data. There are also critical moments and phases of rapid growth, when programmers proceed to code debugging, adding new modules, or else restructuring platform while it is still running. Every platform starts (near) empty, without data. During its life, it of necessity gathers more and more information, which it rearranges and changes to respect evolving situations. Even without assuming intelligence and intentionality, we may be tempted to speak of *learning platforms* to describe this evolution.

Thinking of a platform as open (as different from simply *designing* an open platform) may be the right way to reflect on and further investigate this strange type of evolution.

[18] This simple expression (*added_notes* * 7) + (*message_count* * 5) + (*exercise_count* * 3) + (*history_count* * 2) clearly gives more importance to actions that convert tacit to explicit knowledge (i.e. notes and messages) than vice-versa. It has been proved significant for standard groups (20-25 students) and for medium dimension courses (count of units <100) as a tool to discriminate among «writers» and «readers».

BIBLIOGRAPHY

Ausubel, D. (1968) *Educational Psychology: A Cognitive View*. New York, Holt, Rinehart & Winston.

Collins A., Brown S.J., Newman S.E. (1989) «Cognitive Apprenticeship: Teaching the Crafts of Reading, Writing, and Mathematics», in L.B. Resnick (ed.), *Knowing, Learning and Instruction. Essay in Honor of Robert Glaser*, Hillsdale, N.J., Erlbaum.

Dewey, J., Bentley. A. F. (1949) «Knowing and the Known». In R. Handy and E.C. Harwood (eds.), *Useful Procedures of Inquiry*. Great Barrington, MA: Behavioral Research Council.

Duffy, T.M., Jonassen, D.H. (1992) *Constructivism and the technology of instruction*, Hillsdale, New Jersey, Erlbaum.

Mazzoni E., Bertolasi S. (2005) «La Social Network Analysis applicata alle comunità virtuali di apprendimento», Je-LKS, n. 2, pp. 233-255.

Nonaka I., Takeuchi H. (1995) T*he Knowledge Creating Company*, Oxford University Press, New York.

Novak, J.D., Gowin, D.B. (1985) *Learning How to Learn*, Cambridge University Press, Cambridge, England.

Wasserman, S. Faust, K. (1994) *Social Network Analysis. Methods and applications*, Cambridge University Press.

Wenger E. (1998) *Communities of practice: learning, meaning, and identity*, Cambridge University Press.