Acte Cybercartographique

Peter Pulsifer, Sebastien Caquard, JP Fiset and Amos Hayes

Requirements Gathering

When it comes to computer engineering, the first activity to take place in a project is to gather requirements. In a project like the cybercartographic atlas, this means to interpret everyone vision or dreams of what the atlas might become, and translate those wishes into a set of comprehensive requirements that can lead to a product, in this case the atlas, that will answer the needs that were expressed. As in any human endeavour, it is difficult to reach a perfect consensus; it is much easier to end up with a product that none likes. This is because before a product is visible, everyone imagine it somewhat differently. Therefore, it is important to focus on the needs of the user, and not on the wishes expressed early.

Another difficulty in gathering requirements in this project had to do with the fact it is set in a university geographic department settings. As such, most people are very literate about computers. This is because a lot of emphasis on computers has been brought about with the advent of GIS. Therefore, many persons involved in the project can discuss at great length about implementation details. It would be easy to fall into the trap of taking those wishes as requirements since they are so well articulated from the perspective of a computer engineer. However, the amalgamation of a set of implementation details rarely leads to a product that answers the true needs of the user. In fact, proceeding that way leaves to chance the likelihood of answering the user's needs.

Discussing with team members, one of the recurring theme was to make it easier for potential authors to publish atlas modules. This came from various comments:

- "A student in a school using the atlas should be able to cut and paste some part of the atlas and add his text and then make a report."
- "We want a tool set where we can easily generate the modules we want."
- "In the end, the atlas has to be responsive, where we can click here and get all the information we want there."
- "We will be able to reuse the atlas and publish about many problematic areas where the population should be informed."
- "An atlas wiki is what is really needed."
- "I do not want to learn all that javascript (or other technology) stuff".

All these comments refer to the fact that people find cumbersome to publish material. Geographer have a lot to say, but must struggle with a wide variety of technologies to say what needs to be explained. From those comments, a first requirement was established:

> *The atlas framework must provide a lower cost to publication than other means available at the present time.*

Other requirements that were gathered:

- Atlas content shall be available from a web browser.
- Atlas content should offer highly responsive interactions between the map and the text.
- Atlas content should be derived from live data that reside in distributed databases.
- The atlas should be developed using open standards.
- The atlas should support the research being conducted in the department.

This is not an exhaustive list of all requirements, but it should be enough to illustrate some of the difficulties in design. First and foremost, it is evident that one of the pressing needs is to give easier tools to content authors so that publishing of an atlas module is made easier, and thus hide the technical aspects from the authors. On the other hand, the content shall be highly responsive and provide interactions between the text and the map. The latter points towards a certain level of complexity greater than current technologies offer.

Design

The next phase in development is to design a solution that answers the needs/requirements stated. This is an iterative approach where possible designs are contemplated. At each iteration, current tools are evaluated for fitness and the design is refined. As the design evolves, it is compared to all stated requirements.

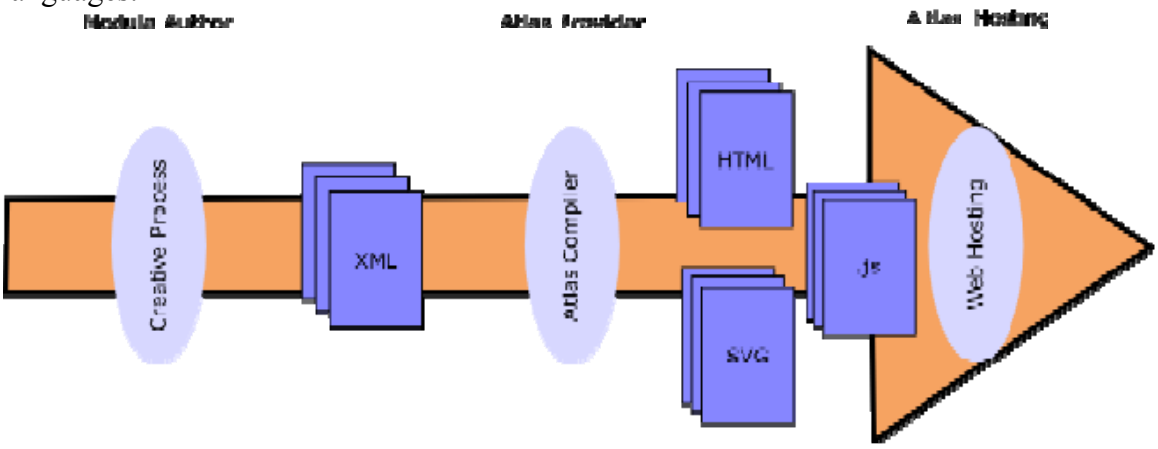
At a beginning of a computer project, it is often difficult to get a handle on the final product. As illustrated in the previous section, users' expectations are very likely to change given the nebulous interpretation of requirements. Furthermore, early prototypes often stimulate discussions as a boundary object is introduced. The prototype becomes a vehicle of understanding between the designer and the customers. Because of this likelihood of change, a good design is one that take into consideration what will change.

In the end, it was decided to take an approach where the atlas modules would be written independently of the atlas' implementation details. Using this approach, the authors of atlas modules have a certain area of responsibility, while the atlas provider and atlas host have other clearly defined roles.

This model was inspired from another open source project called DocBook, where technical documents are written independently of the formatting. In DocBook, documents are written void of formatting details. Instead, a number of tools turn the information into web sites, PDF document or PostScript files (DocBook is explained at http://docbook.org)

This approach allows authors to work on modules, at the same time as the atlas "look-and-feel" is refined. As it is easy to predict that the look of the atlas (its rendering, style and functions) are going to change, it is a good design decision to shelter the work of authors from the implementation details of the atlas. It also partially fulfills the requirement where authors should be presented with a simpler perspective to publishing, one hopefully void of implementation details such as details required by scripting

languages.



The previous figure shows the three perspectives:

- Module Author: The module author is left with the responsibility of creating XML documents that describes an atlas module. This document contains all the text that the author wants to publish. It also specifies the maps that should be displayed. And finally, it contains markup elements that links features on the map to information found in the text.
- Atlas Provider: Responsible for transforming the work of the authors into a set of packages that can be published on the web. The atlas provider makes decision about the organization of the pages, the look-and-feel of the pages, the interaction level between the map and the text. The atlas provider produces a set of web files, including HTML, SVG, Javascript and Servlets that can be deployed on a web site.
- Atlas Hosting: In our model, this is the last step. The activity of atlas hosting entails to make available all of the atlas provider packages with all the supporting applications and databases.

This organization takes away a lot of the rendering aspects away from the module author. Because atlas modules are given a “cookie template”, then the author can concentrate on the text and the maps without worrying about technology decisions that are affected by the technological issues. On the other hand, the author is constrained to the facilities offered by the atlas provider when it comes to displaying the maps.

At this point, XML documents are still too technical for authors. We are hoping that with time, tools will be built to help generate those documents. Easy to use graphical user interface applications would probably be the best suited for the needs of the authors.

The atlas provider is left with the task of weaving the different modules into one coherent atlas. Tables of content and navigation systems are required to proceed further. At this time, there is only one ‘atlas compiler’, since we are devising a prototype. Our hope is that with time multiple incantations of compilers would be available. Therefore, a module could be reused in multiple atlases.

Implementation

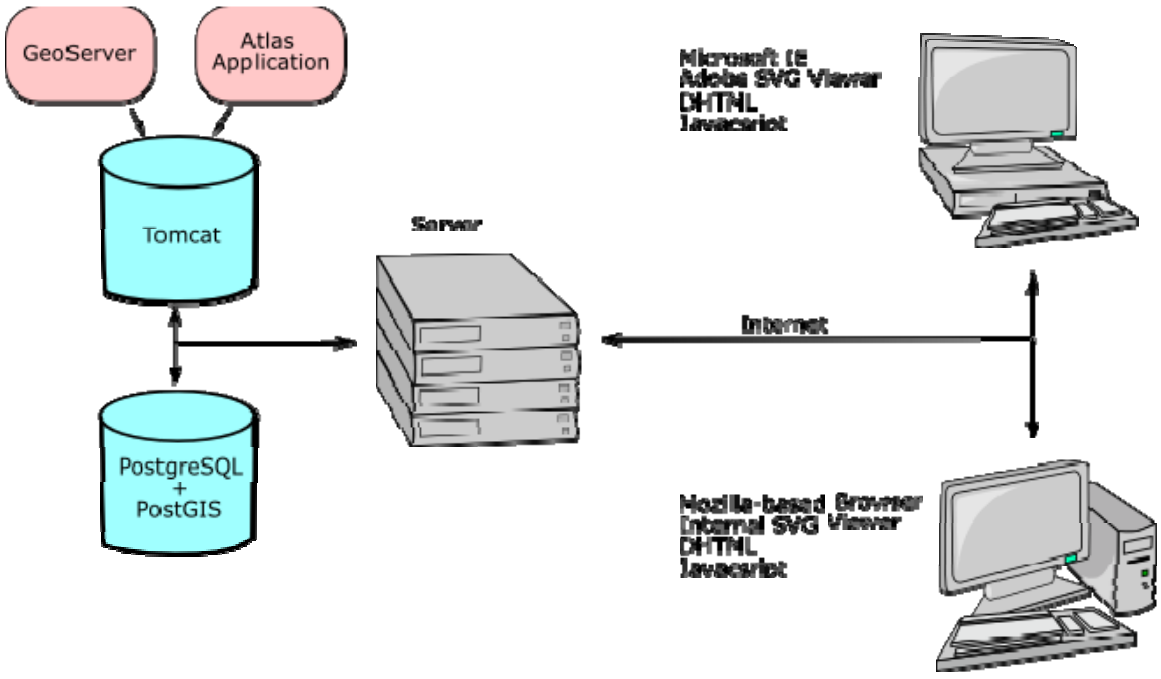
Once design has been agreed upon, implementation takes place. The design and implementation phases are never completely dissociated since implementation details often affect the overall design. Often, early prototypes are assembled to prove or disprove some design assumptions. Also, review of available projects that could potentially be reused can often influence the design phase.

Many technologies and projects are reviewed, and here is a list of technologies that were considered:

- GIS: MapServer, GeoServer, Deegree
- Databases: PostgreSQL, MySQL
- Web Servers: Apache HTTPD, Jakarta Tomcat
- Client Applications: Microsoft Internet Explorer, Mozilla, Adobe SVG Viewer, Macromedia Flash
- Browser Technologies: HTML, DHTML, Javascript, SVG, Flash
- Countless of available tools and Open Source projects

Each technology is considered against a number of objectives such as ease-of-use, compatibility with other technologies, deployment facilities, level of industry adoption, and many more.

Finally, a set of technologies/projects were selected and an implementation plan was established. The following figure shows the technologies that were selected, and how they interact. This is basically a detailed view of the "Atlas Hosting" perspective, introduced in the previous figure.



Conclusion

It is simple to lose the voice of the cartographer in the production of a project such as the one described here. From the perspective of a computer engineer or scientist, the elements of cartography form only one part, although an important one, of the whole story. As explained, the requirements definition has many human challenges, while the design and implementation phases face many technical barriers.