

A new approach to On-line Mapping

Blanca Mancilla and John Plaice

School of Computer Science and Engineering
University of New South Wales
UNSW Sydney NSW 2052, Australia
e-mail: {bmancilla,plaice}@cse.unsw.edu.au

Abstract

Internet mapping and the widespread availability of mapping software are revolutionizing the world of cartography: anyone can now produce maps of many areas of the world. However, these tools typically only allow one to produce pictures, and the process undertaken to achieve these pictures is lost.

We present a new approach to cartography in which maps are understood to be the visual representation of a multidimensional context, and the space being mapped is one such context. This approach to mapping – called *intensional* – allows the developers of maps to keep track of all the choices that are being made, whether visual, structural or socio-historical, and to share these choices with others, in real-time. As a result, electronic maps become much more flexible and versatile, and much more supportive of speculative research.

1 What is a map?

A map is the visual representation of the ground or its environment. A hard-copy map is a stand-alone document that carries little or no history of its development and the supporting data is not attainable. A user viewing such a map, has to view the representation of the ground solely through the eyes of the cartographer with fixed settings.

The introduction of electronic mapping has

somewhat changed this situation. Certain Internet mapping tools [3, 5] allow the user to manipulate a fixed set of data, allowing control of a restricted number of parameters. Still, in the end, the user is left with a graphical file, containing just visual information. No further manipulation is possible on this file. Furthermore, in some cases the user might not be able to reproduce it on demand. The supporting data and history associated with its evolution can disappear. The main difference with the hard-copy map is that the user can create his or her own map. As trade off, electronic maps often do not achieve the quality of the best paper maps, although this problem will disappear with further developments in graphics software and hardware.

In general electronic atlases today manipulate a fixed number of datasets and allow variations of a restricted number of parameters, producing a given number of possible maps. Extreme cases use only one dataset and/or most maps are precalculated, reducing the choice available to the user.

Commercial GIS [1] tools are in a different category for a few reasons. First, they are reserved to a restricted subset of the population because of their high cost (proprietary software). Also, they are to be operated only by *experts* as the user interface can be very complicated. Friendly interfaces have been built that perform only dedicated tasks, unreachable once more from general use. On the positive side, their manipulation of data is im-

proved, and they allow the use of a few different datasets, even user-crafted, for mapping purposes and extra processing. However, the use of each dataset must conform with the particular GIS standard and its use must be explicitly specified.

The ideal e-map would facilitate the following:

Scenario I The user wants to manipulate the same dataset and view it using different perspectives. After a series of iterations, the user decides she is happy with the final product. The mapping tool she used should keep track of the computational environment at each iteration and of partial results. The user might decide to recalculate or change the final output using an intermediate result.

Scenario II The end result of scenario I consists of more than one map. It is a *family* of maps, (and the user needs them for comparison, visual linkage, or to point out differences). If the user decides there is a qualitative change on a parameter that needs to be reflected in all the versions, this should happen seamlessly. If on the other hand the change is to apply to a few versions only, the system should handle this as well.

Scenario III The user wants to manipulate different datasets and create a family of maps. Adding a new dimension to scenario II, say `dataType`, which varies in the number of available datasets,¹ might lead to the creation of many more new maps. It is to be noted that the number of datasets can be infinite and adding a new one does not change the basic implementation. Conceptually, all these maps exist; physically, they are to be created on a demand basis. The user should not see the difference between the two. What the user can do with the maps should be

flexible and the GUI should reflect this. It should be possible to browse the dimension and dimension-values of both the structure and the content of the document; change the lay-out of the screen, colours, etc.

How would today's tools handle these situations? Scenario I is feasible by creating system stamps – that is a list of the system values – for each relevant instance. The system has to create a mechanism to order and render these stamps on request. As the number of stamps increase, their efficient handling becomes more complicated and with existing techniques, the system performance will deteriorate.

For scenario II, the system must hold the family of maps. Each version of the map is treated as a separate entity that is not linked to the others. A map that has not been produced does not exist.

For scenario III, the tool might limit the variation in the data and fix the number of new maps to create. These are just a few of the problems, and the solutions are not satisfactory.

Ideally, a user should be able to zoom in a map and as the level of detail changes significantly, a different dataset should be used. For example, when viewing a map of Asia, the user might want to look closer at a particular location. When at a country level, the detail seen is different from the continent level. Now the user is viewing a map of Japan, but wants to get a closer look at the Tokyo area. What is seen at this level should be coming from a different dataset than from the continental level. If the process continues all the way down to a house level, the plan of the house should pop up.

Some of today's tools allow the user to zoom in by clicking on a map. Since most of these use only one dataset, after zooming repeatedly the user finds empty space with isolated elements. This is *not* a representation of the ground. Encarta [2], for example, allows

¹For example, if there are five datasets, the dimension `dataType` can have any of the values [1, 2, 3, 4, 5].

the user to zoom from world scale to street scale in your home town. At first, this appears innovative. But the user does not have many options on how to create the map. Most of the parameters are fixed: The system does not offer the ability to change the colouring, choose the language or add a new dataset, to name just a few aspects.

1.1 A new definition of a Map

An electronic map is a flexible document, that allows the user to manipulate its structure as well as its content. An electronic map is really a family of versions of maps and should be perceived as a whole. Each version is an *instance* of the map. The map is specified by parameters that define its structure, its content, its output representation, as well as the underlying environment in which it is calculated.

The behaviour and functionality of a map is dependent on the values that these parameters hold. As the map becomes a more complex entity supporting a more complex set of features, so do its parameters, their relationships and their manipulation.

What is described in the previous paragraph is by no means easy to achieve with existing technology. It becomes necessary, then to develop a theoretical framework for managing the complexity of map parameters. The approach that best handles such complexities is the Intensional framework, where every parameter is viewed, mathematically, as a dimension placed in a multidimensional space. Every point in the space encapsulates a context in which the map currently exists and thus inflicts a particular behaviour on the map under that context.

These dimensions have a full interdependency relationship; dimensions can be orthogonal, nested and/or dependent on others. The framework should allow for dimension values to change or new dimensions to be added, as was the case for scenario *III*.

Tools based on the formal theory for mul-

tidimensionality are called intensional tools. Maps are multidimensional entities, and as such they should be manipulated using such tools.

2 Intensional Programming

Intensional programming [8, 6] is an approach to programming in which programs are assumed to run in an implicit multidimensional context, which is a point in a multidimensional space. The space can be in theory arbitrarily complicated, but a simple Cartesian space, where the dimensions and values are simple scalars (strings and integers), is often enough.

Formally, an intensional program is understood to be a function from the multidimensional space to the domain of ordinary programs. Informally, the program is understood to be immersed in the context. When it needs to, the program can test the current values for one or more dimensions and choose alternate paths according to these values.

This process applies not only to programs but to all entities that can reside on a computer: programs, components, data, Web pages, databases, etc., can appear differently at different points in the multidimensional space. If a dimension is irrelevant to one of these entities, then it is nowhere mentioned with respect to that entity: this approach allows the space to be infinite-dimensional without in any way affecting clarity or performance.

Languages based on intensional programming have been created and are being used successfully in a number of different situations [4, 7]. These languages have been used to build a number of interesting applications, including multilingual distance education and multidimensional documents. They are easily applied to other situations, and it is conceptually not difficult –although probably not easy in practice– to add intensionality to all sorts of applications, including file systems, proto-

cols and operating systems. In addition, there are means for sharing multidimensional contexts among processes, using what are called *aether servers*.

3 Interface and mapping tool

Using the generic mapping tools, GMT [9, 10], one of the authors (Mancilla) has created an interactive mapping tool for the Web, using, as is standard, CGI scripts in Perl, and HTML forms. Choices are currently limited to a dozen options; clickable maps are currently under development. The current tool resembles other tools found on the Web. But with any of these tools, it is difficult to take full advantage of GMT capabilities, since good mapping tools provide myriads of parameters, or dimensions, which current technology does not handle naturally. The authors of GMT purposely created the tools without a graphical interface, using the command line as input: they did not want to introduce a handicap in the power and functionality of their system.²

Modification of the current interface requires changes in—at least—three separate places and adding a substantial amount of code. Maintenance is complicated. Using an intensional language should largely simplify the task of updating and enhancing the code because the manipulation of dimensions is inherent in context-aware languages.

At the functionality level, keeping track of the workflow process is crucial. The present interface requests the user for input parameters, processes them, and on error—sometimes—returns a message, otherwise outputs a *beautiful* map. The map is, as mentioned in a previous section, a graphical file impossible to manipulate as a map. In addition, if the user wants to see the parameters entered, he can press the back button, standard in Netscape or similar. But after creat-

ing a series of maps, the user would be unable to understand the workflow, some of the information would be lost, the real sequence would have disappeared. But from the intensional point of view, the workflow is simply a sequence of contexts, all elements of a multidimensional space. Elements across workflows might be linked, since they are just points in the multidimensional space, which forms a lattice-like structure. The session log takes on meaning, and can be manipulated, shared, analyzed and dissected as needed.

In keeping with the previous paragraph, a map corresponds to the final point in a sequence of contexts. It is the workflow that led to it, the session log where each instance is described by a context. The session log allows the user to answer the question, “How did you get to this point?” Furthermore, the user should be able to *annotate* sections of the workflow. For example, from `context(A)` to `context(B)` the user applied a filter, $f(x)$. The user could annotate the filter by adding a comment on why this decision was made and explaining technicalities.

With the concept of context and session logs, the notion of *sharing* becomes a real possibility. Sharing here refers to collaborative work, people getting together to produce something. Or sharing of points of view for discussion. This sounds like the sort of thing that should happen in any research environment, although it typically only happens at the personal level, with the computer as just an annex.³ Why shouldn't people get together in different corners of the world via cyberspace to work on a project together, or discuss ideas and points of view? In other words, why not use the computer to facilitate people's interactions?

Let's refer specifically to mapping. An infrastructure of this kind would allow different users to create contexts that other users could share in order to create the different maps.

²The down-side, GMT could be difficult to use to its full power.

³Whether this happens or not, is not part of the discussion of this article.

Users could contribute to the definition of the present context for everybody else to share the results. Or one user could lead a tour through a series of contexts while the other users follow, a situation similar to lecturer and students.

Another possibility is where users individually build a map and from the result a conclusion is drawn. But when different users share these conclusions, they realize they do not all agree. The infrastructure should allow users to *view* other users' contexts, and bring them into their own working space to manipulate them with their own settings. This would allow them to discern the points of disagreement and that done, users could focus on them. By using parametrization – the use of dimensions– users could agree on a fixed frame for the parameters they do not care about, and work on the rest. It is like putting two things in parallel and doing an eye scan for discrepancies. Once only the points of disagreement stand out, resolution should be easier. Using the recorded session log, users can go collectively through the process and explain each point with clarity instead of claiming magic.

4 Conclusions

Viewing maps through an intensional lens is very promising. Already our limited prototype for interactive mapping allows us to do a number of tasks that are difficult to achieve by all but some of the most expensive commercial tools. As the ideas in this paper are developed, research in any discipline that necessitates maps will be substantially easier.

References

- [1] *ArcInfo*. <http://www.esri.com/software/arcinfo/>
- [2] *Encarta*. <http://encarta.msn.com/eng>
- [3] *The Map Machine*. <http://plasma.nationalgeographic.com/mapmachine>.
- [4] Gordon D. Brown. *Intensional HTML 2: A practical approach*. Master's thesis, Department of Computer Science, University of Victoria, Canada, 1998.
- [5] Xerox PARC. *Map Viewer*. <http://mapweb.parc.xerox.com/map>
- [6] John Plaice and Joey Paquet. *Intensional Programming I*, chapter Introduction to Intensional Programming. World Scientific, 1996.
- [7] P. Swoboda and W. W. Wadge. Vmake, ISE and IRCS: General tools for the intensionalization of software systems. In *Intensional Programming II*. World-Scientific, Singapore, 1999.
- [8] William W. Wadge. *Intensional Programming II*, chapter Intensional Logic in Context. World Scientific, 2000.
- [9] Paul Wessel and Walter Smith. *The Generic Mapping Tools*. <http://www.soest.hawaii.edu/gmt>
- [10] Paul Wessel and Walter Smith. New, improved version of generic mapping tools released. *EOS transactions*, 79:579, Agu 1998.