Specifying Open GIS with Functional Languages

Andrew U. Frank
Werner Kuhn

Department of Geoinformation
Technical University Vienna
Gusshausstrasse 27-29/127
A-1040 Vienna (Austria)

Frank@geoinfo.tuwien.ac.at
Kuhn@geoinfo.tuwien.ac.at

Abstract

The concept of Open GIS depends on precise definitions of data, operations and interfaces. This paper argues for the use of functional programming languages as specification and prototyping tools for Open GIS components. It shows how functional programming languages fulfill the key requirements for formal specification languages and allow for rapid prototyping in addition. So far, it has never been possible to integrate specification and prototyping in a single, easy to use environment. Most existing specification methods lack appropriate tools for checking and prototyping, while existing tools lack either sound semantics or usability or both. The paper discusses the role of specifications in GIS, requirements for specification languages, and presents the basics of algebraic specifications as well as of functional languages. It then describes how functional languages can be used for writing and executing algebraic specifications. A brief example of a GIS data type specification in a functional language is presented, showing how specifications serve to describe differences in the semantics of GIS operations. We conclude that functional languages have the potential to achieve a breakthrough in the problem of specifying interfaces of interoperable components for Open GIS.

1 Specifications and Open GIS

Specifications are essential for software quality and are widely used in industry. For Geographic Information Systems (GIS), they are of special interest in the standardization of data models, transfer methods, Open GIS component interfaces, and database architectures. Practical specification methods are needed for the success of Open GIS architectures in which programs and data collections from different vendors need to cooperate at multiple levels of abstraction [Voisard, and Schweppe, 1994].

1.1 Specifications and Software Development

Specifications allow for a division of labor in software development. They serve as a contract between the software analyst, who understands the application problem, and the programmer, who is concerned with an optimal use of resources. They support producing the software and assessing its correctness.

CASE tools provide informal methods to design, manage, and communicate specifications for software. Positive results from largely informal software design

1 Funding from Intergraph Corporation and from the Austrian Science Foundation is gratefully acknowledged.
methods in general have been reported [Head, 1994], but software practitioners criticize current tools for being too low level. A pointed observation is that CASE tools are nothing but glorified systems to draw diagrams and often do not scale up to large problems, where the amount of documentation becomes overwhelming.

Formal specifications, i.e., specifications written in a formal language with mathematically defined semantics, allow for formal checks and reasoning before programming starts. They can provide support for an automatic program verification [Guttag, Horning, and Wing, 1985]. However, such consistency checks are internal to the formal system and cannot ensure that the specifications capture the intentions of the designer or other real world requirements.

Rapid prototyping has been advocated to achieve programs that correspond not only to specifications but also to the actual user requirements. Prototyping reduces the danger that you get what you ordered, but not what you wanted. In current software design practice, specification and prototyping tools are often separated or only loosely coupled. If a language allowed to write specifications and to execute what has been specified, it could serve as a combined specification and prototyping tool. This paper claims that functional programming languages, extended with recent research results, achieve this goal.

1.2 What is Special about Specifications for Spatial Data?
The need for formal specification languages is common to the whole software industry; problems with interoperability of tools from different vendors plague nearly every user of a computer system. However, the specification problem is more acute for GIS than for most other application areas:

- Economic use of spatial data is only possible if data can be used by many different users from different organizations. The need for a functioning market of spatial base data is larger and this market is growing faster than in most other areas of information technology. Consider as an indicator the offerings for data on CD-ROM: geographic data figure very prominently.

- The structure of geographic data is more complex than that of other data exchanged routinely. It is comparable to, but going beyond, that of CAD data, where similar problems are encountered.

- Sharing data transcends organization boundaries. The consumer usually pays for the data and expects them to be "usable", otherwise legal problems can ensue [Frank, 1992].

1.3 Specifications and the GIS Market
GIS are evolving into multi-vendor software environments, Open GIS, where heterogeneous components cooperate to solve complex spatial problems. Users buy software from different vendors to solve particular parts of their problem: they want to access the database system produced by one company from the mapping tool of another vendor or to assess the spatial dimension of marketing forecasts in a GIS. This movement toward interoperable, open environments will rapidly progress within and outside the GIS industry [O'Callaghan, 1993]. Specifications are the key to achieve this interoperability. They constitute a contract between providers and customers of services in an Open GIS.

Specifications of GIS services provide economical viability for the niche market vendor who can offer tools that are independent of the software environment of a