2010

Application Planning

Jochen Albrecht
CUNY Hunter College

Clare Davies

Chapter 7 – Application planning

Jochen Albrecht, Clare Davies

One of the mantras of software development is the need to always keep the application's user in mind; it is she that all our efforts are for. This is the core principle of User-Centred Development, as we have discussed in Chapter 5. We need to understand what it is that the user is trying to accomplish, what her experience or skill level is, and how the task to be solved fits into a larger workflow and purpose. As such, in UCD application planning is very much like systems analysis where the different elements of the system are carefully designed, except that here the user is at the centre of it all. One of the core differences between general information technologies application and traditional GIS applications is the steep learning curve that is required from the users due to the complexity of the software and the fact that this complexity is exposed in all its glory (or ugliness) on the interface. Even when it comes to other geotechnologies, such as PND or web-based mapping, beyond some basic operations it is common to see that the system is becoming complex to operate, requiring the user to learn several disjoint steps to accomplish a specific task – such as finding a hotel. This chapter's first two sections discusses some of the issues of GIS interface and the ways researchers address this issue, and we focus especially on task analysis. While Lanter and Essinger (1991, p. 1) noted that "some of the most successful user interfaces are complete illusions outwardly bearing no resemblance to the data processing happening inside the machine" almost two decades ago, most GIS interface – from commercial to open source - require the user to have significant understanding about the way that the system is implemented to operate it. Users have mental models about the tasks they accomplish with a system – this is their internal representation of what they want to do, what the computer does and the way the system lets them accomplish those tasks. These models are defined by the user's prior experience, existing knowledge, and preconceptions about tasks (Rauh *et al.*, 2005). For both a task and the way to accomplish the task to make sense, they must correspond to existing knowledge the user already has. Section 3 will dwell on this transition between the developer's view and the user's view of a GIS application. Formal specifications of both views are seen as one possible way to create a consistent user interface, and can be used in the design process – especially of major systems where there is a large team of developers and there is a need for formal and accurate definition that the whole team can use. For many years, researchers have been gathering user requirements, but as section 4 alludes to, this is becoming an increasingly difficult with the

proliferation of contexts in which geotechnologies are use, and especially with the emergence of user generated content web sites and compositions of web services through mash up. Clues may be drawn from another domain, that of workflow management in the business community as well as in scientific process composition. Sections 5 and 6 discuss different approaches in these two domains. We laso describe the importance of the research body on GIScience ontologies which has some very practical value in GIS application development based on the semantic web. Recent efforts in this realm are presented in section 6, while the remainder of this chapter provides pointers what technical and scientific developments to watch.

7.1. GIS interface complexity

As noted in Chapter 1, GIS combines the functionality of database management systems and computer aided design software, with cartographic design and statistical analysis added for good measure, so it should not be surprising that it is one of the most complex widely used applications around. Not only does it typically consist of over a thousand functions but it requires the user to combine different metaphors and concepts borrowed from separate disciplines in order to accomplish even a basic task (Driver and Liles 1983; Mark and Gould 1991; Traynor and Williams 1995). In consequence, it usually takes a combination of multiple courses at university level and years of experience to become a proficient user. A lot of this has to do with the multi-pronged origins of GIS itself, which was developed by surveyors, landscape architects, and census statisticians, to name three prominent origins. These disciplines have little in common but contributed approximately equally to the conceptualization what we now know as GIS. The National Center for Geographic Information and Analysis (NCGIA) conducted a series of workshops in the early 1990s that brought together experts from a variety of disciplines to tackle the complexity of GIS user interfaces (Frank 1993, Nyerges 1993; Turk 1993, Volta and Egenhofer 1993, Medyckyj-Scott and Hearnshaw 1993). In this early literature, the issue of understanding and identifying the tasks that the user perform with the system was recognized as central to the design of well functioning and usable GIS. This is based on the understanding that unless we can clearly express what are the tasks that the user is trying to perform with the system we cannot develop a system that supports this work. Importantly, the tasks are not about what the user will do with the system, but rather what are the real world transformations that should take place. In most cases, the computerized system will support some task that the user can perform without the use of computers, and it is this type of activity that we want to capture. For example, the most basic operation with a GIS is ‘create a map’. This can be done without computers, but require careful drafting and ability to trace base maps, transpose their co-ordinate systems, etc. Thus, the use of GIS makes such a task faster than carrying it out manually. However, if we are setting out to analyse the user’s needs, we can develop a task description that focus mostly on what the user is trying to achieve without any consideration to how the system will be implemented.

Task description and task analysis are, therefore, central to the process that we describe here of hiding the complexity of the system from the user.

7.2. Task analysis in GIS

There are many well-documented methods of describing work tasks, both within and outside the context of computer use (Kirwan and Ainsworth 1992). Most of these methods focus on the qualitative or quantitative description of tasks, knowledge requirements and errors, without any specific methodology for extracting prescriptions about individual actions, for example differentiation between tasks that directly contribute to the outcome of the task (selecting the color of a line on the map) and enabling actions, such as switching the computer on. An exception is the task-action grammars (TAG) proposed by Payne and Green (1986). TAG splits a task into individual actions, mostly described using terms that suggest a preparing/performing distinction (e.g. `select object' , `modify'), but the grammars are aimed more at uncovering inconsistencies in the patterns of actions than at identifying individual actions as productive.

This emphasis on analytical description is not surprising, given that task analysis may be performed for many different purposes, ranging from risk calculation to the design of training courses. The prescriptive requirements naturally vary with the goals of the analysis. The work/enabling distinction proposed by Whitefield et al. (1993) should make it possible to identify portions of user-system interaction that involve many enabling actions or a particularly long time or high amount of effort spent on such actions, for each work action. As a result, specific aspects of the interaction may be identified which would particularly benefit from a redesign if we are examining an existing system or evaluating a prototype; the purpose of such an exercise may be either to reduce enabling actions, to reduce or alter them for some users whose expertise or tasks cause the actions to be unnecessary or tedious, or to change the nature of the enabling actions so that they more closely support the user's intentions within the task. Ultimately, this is just one more tool intended to create systems closer to users' real needs.

Existing analysis methods already allow us to quantify work using the number, duration and apparent physiological workload of actions, as well as the percentages of these spent in making or recovering from errors (e.g. Zapf et al. 1992).

Task analysis can be linked to ideas which arose much earlier in the history of analysing work. Any book on time management, of which there must be hundreds, will attempt to show the readers that they spend vast amounts of their time on non-productive activity. It is probably true that no form of human activity, from cookery to librarianship, is free from what Whitefield et al. (1993) labeled `enabling' behavior. Meetings, completing forms, searching for keys and placing paper in a printer tray could all be viewed this way. We should be cautious in assuming

that the presence of enabling actions within a computing task is in itself a signal to redesign the system: it may be that we can never eliminate, nor with to, some enabling actions from computing tasks. .

The existence of a variety of actions and procedures within a task may in fact benefit workers' or personal user's motivation under many circumstances, as was demonstrated in the revolt against Taylorism and the move towards job enlargement/enrichment earlier in this century (Kelly 1982). Taylorism, a way of thinking which centered on the improvement of industrial productivity through the implementation of `the one best way' to perform each task, became outmoded as work analysts began to focus on the social aspects and non-financial motivations of the workplace. Greif (1991) compared Card et al.' s (1983) GOMS and Keystroke-Level models, which were covered in Chapter 6, with Taylorism in their assumption that it was possible to identify the one `best' , most efficient, method of performing any given task. Greif (1991) argued that such assumptions ignore the importance of considering human motivational needs, especially the need for `growth'. Current interest in user experience, which extends the analysis beyond efficiency per-se, can be seen as a way to include these motivational needs.

According to Greif (1991), German work psychology, and particularly the work on `action theory' , favors systems which are adapted to an individual users' understanding of their tasks; thus the choice of available actions and the level of detail inherent in those actions should be made to fit the individual's conceptual model. Those conceptual models in turn, according to the German theorists, are based on users' motivational needs. The motivational need for `growth' and fulfillment does not mean that all non-goal-achieving activity is desirable; nor does it prevent us from identifying such activity even if we decide not to remove it. The work/enabling distinction is related to the concept of `idle time' in production lines, which consists of balance-delay time (time spent waiting for another person or system to complete a process), non-productive time (time spent handling or moving materials, or in checking errors and defects) and waiting time (time wasted due to machine breakdowns or interruptions in supply) (Kelly 1982:71). Such periods are clearly non-productive, and the worker in a poorly designed production line has no choice but to `perform' the waiting actions in order to continue the task. Additionally, some method study techniques distinguish within an operation between actions which `make ready', `do' and `put away' (Kanawaty 1992:95), of which the first and third are clearly analogous to `enabling' actions.

However, care should be taken in analyzing office-based work with techniques developed for industrial production: often the aims of the techniques are different as well as the context of use (e.g. they may aim to identify a standard production rate or assess workers' personal performance). Additionally, in manufacturing contexts workers' activities are largely physical and thus subject to physical limitations. Thirdly, the goal of the activity is always clear: making

(or physically altering) something. Finally, the worker has not chosen her or his goal: the task is fixed and usually does not involve mental creativity.

Nevertheless, even in the domain of creative work within office-based computing tasks, Green (1990) proposed a conceptual `dimension' that may be linked to the work/enabling distinction. This is the `cognitive dimension of viscosity'. Green defines Viscosity as the degree to which a system makes it difficult for a user to change her or his mind - the difficulty of altering an entity after it has been created. Thus a program written in a disorganized way could require extensive rewriting if the programmer later decided to incorporate extra functionality; a road marked within an image editing package ('raster GIS') may not be easily changed if the user decides to alter a junction. Again, the viscosity dimension has limited applicability - not everything one does is creative, and not every source of inefficiency is due to difficulty in altering created artifacts – but nevertheless the extra effort expended in a highly `viscous' system certainly entails `enabling' activity and is arguably unnecessary.

Another conceptual framework for tasks was offered by Donald Norman in his exploration of `stages and levels' in HCI (Norman 1984). Norman's four-stage concept of a task meant splitting it into intention, selection, execution and evaluation. Recognizing that the execution of a task to fulfill a given goal was usually likely to involve a complex sequence of actions, Norman (1984) argued (at a time when most computer users were not yet using graphical user interfaces) that systems requiring the user to `point' to select an action (practically, a GUI-based system) were likely to be more efficient, but more restricted, than those requiring the user to name the action, for example by typing its command name. Norman (1984) also went further and argued that the difference between users who preferred GUI systems such as menus, and those who preferred `name'-based systems such as command-line user interfaces, was to a large extent attributable to the emphasis placed on the selection of actions as opposed to their rapid execution. This is turn, he argued, reflected the type and extent of a particular user's knowledge: some users knew all the possible actions already, and wished to execute one as rapidly as possible; others wished to be visually reminded of what it was possible to do. In the work/enabling terminology, one could argue that the emphasis of the latter group was on being supported in enabling (selecting) the action, while the former group focused on minimizing enabling actions and just working (executing). Both the work and enabling actions within a given task will fall largely within Norman's (1984) third `stage' of a task, i.e. the `execution' stage, although some will be part of the earlier stage of `selection' or the later stage of `evaluation'.

Many of these task analysis frameworks were proposed for systems were tasks are fairly easily identified and defined, such as word processing. In contrast to these systems, GIS are sophisticated products, with some of packages place more than 10,000 commands or functions

at the user's disposal. Nevertheless, the most common tasks performed in reality are more mundane ones such as preparing maps for printing or plotting on paper, querying and updating database records, data entry by digitizing the map data, checking for and correcting errors, searching through the system to find particular information, and summarizing data (such as sales totals or highway accident statistics) into printed reports. GIS differ from text editing software not just in the type of data handled, but also in their use for information storage, retrieval and analysis as well as editing. Any attempt to improve GIS usability must start with the recognition that users interact not only with a computer system but also with a map, and thus the analyst should be familiar with cartographic literature on map design and use as well as ergonomic prescriptions about the user interface (Wood 1993). Maps, and hence the software to handle them, require decisions to be made and recorded about projection, scale, symbols, colors, relief representation, generalization and labeling. With this added complexity, for the task analyst as well as for the user, we should not be surprised that little published work has analyzed GIS tasks in detail.

Some authors (e.g. Nyerges 1993, Gould 1994) have called for the gradual development of detailed task taxonomies for GIS, but to be feasible such taxonomies would have to develop from a relatively high-level definition of `task'. Nyerges (1993) adopted a view of GIS tasks based on the idea of tasks as transformations, and the `transformations' in question concentrated on the transferral of information between mental, real-world, graphical and digital forms. It could be argued that such changes of form constitute a subset of the transformations a user may have to perform within the overall work domain.

Albrecht (1994) argued that the GIS user faces a fundamental problem in having to translate high-level job requirements into long sequences of low-level system-specific commands. Therefore, he proposed the development of a high-level language for translating between these two task levels, to facilitate GIS use. While this idea focused on the use of GIS to solve analysis, modeling or decision-support problems, the paper also included a genuine attempt to present a hierarchical breakdown of sample GIS-related tasks. This notion of a high-level `language' is one possible solution to the problem of simplifying user system interaction, and the design of such a language could be facilitated by applying a task analysis coupled with an evaluative tool such as the work/enabling distinction. Other possible ways of describing or taxonomizing GIS tasks have been discussed in Davies (1995).

At a much finer level of detail, Haunold and Kuhn (1994) used the GOMS `Keystroke-Level Model' of Card et al. (1983) to analyze manual digitizing of property data. Their study illustrates the usefulness of such a low-level analysis in identifying potential productivity improvement, through the reduction of the number of actions necessary to fulfill a low-level task goal. By way of an example, Haunold and Kuhn (1994) described the steps required in the system they tested

to enable the user to zoom in to an area of the displayed map, and suggested that the ideal number of actions would be only two: a single button press by the operator and the system processing time required for redrawing the screen. Not all operations are as obviously open to redesign as that zooming action; in addition, when faced with a rich assortment of operations it is not clear where an analyst should focus attention to maximize gains in productivity. The potential value of the work/enabling distinction lies in its distinction between the actions that really do the job, and those which do not, a distinction made implicitly by Haunold and Kuhn (1994) in their zooming example. Making this distinction may allow us to see which tasks have a relatively high proportion of enabling actions, and decide whether these are the tasks that would benefit most from redesigning the user system interaction. In other words, distinguishing `work' from `enabling' may add a prescriptive dimension to the descriptive procedures of task analysis.

Case study 7.1 Development of a land use-based transportation model

Albrecht et al. (2008) describe a rather complex but fairly traditional GIS workflow (Figure 7.1 depicts the program's logic; the database schema is too deep to be represented in a single figure). Goal of this spatial decision support system (SDSS) is to model existing flow in a multi-modal network (that is, a system in which people are using several modes of transport such as car and public transport) and to find optimal locations for transit hub based on existing parcel-level land use. The procedures of the SDSS combine many hundred GIS operations for a single model iteration.

The very model *structure* is dynamically changing as the model is run. For instance, the choice of a commuter's mode or route is influenced by how many other commuters crowd one's first or second option. Many commuter train riders, for example, are familiar with the added number of passengers when flooding closes roads during rain or snow storms. The same happens on a smaller but cumulatively as effective scale every day in metropolitan areas around the world. From an abstract perspective, the phenomenon is well studied and has become a popular example in complexity theory. From a user interface development perspective, it is almost impossible to create an easy to use and intuitive interface that hides this complexity from the user. Of course, sometimes it is quite useful, especially for the scientifically inclined user, to see this complexity.

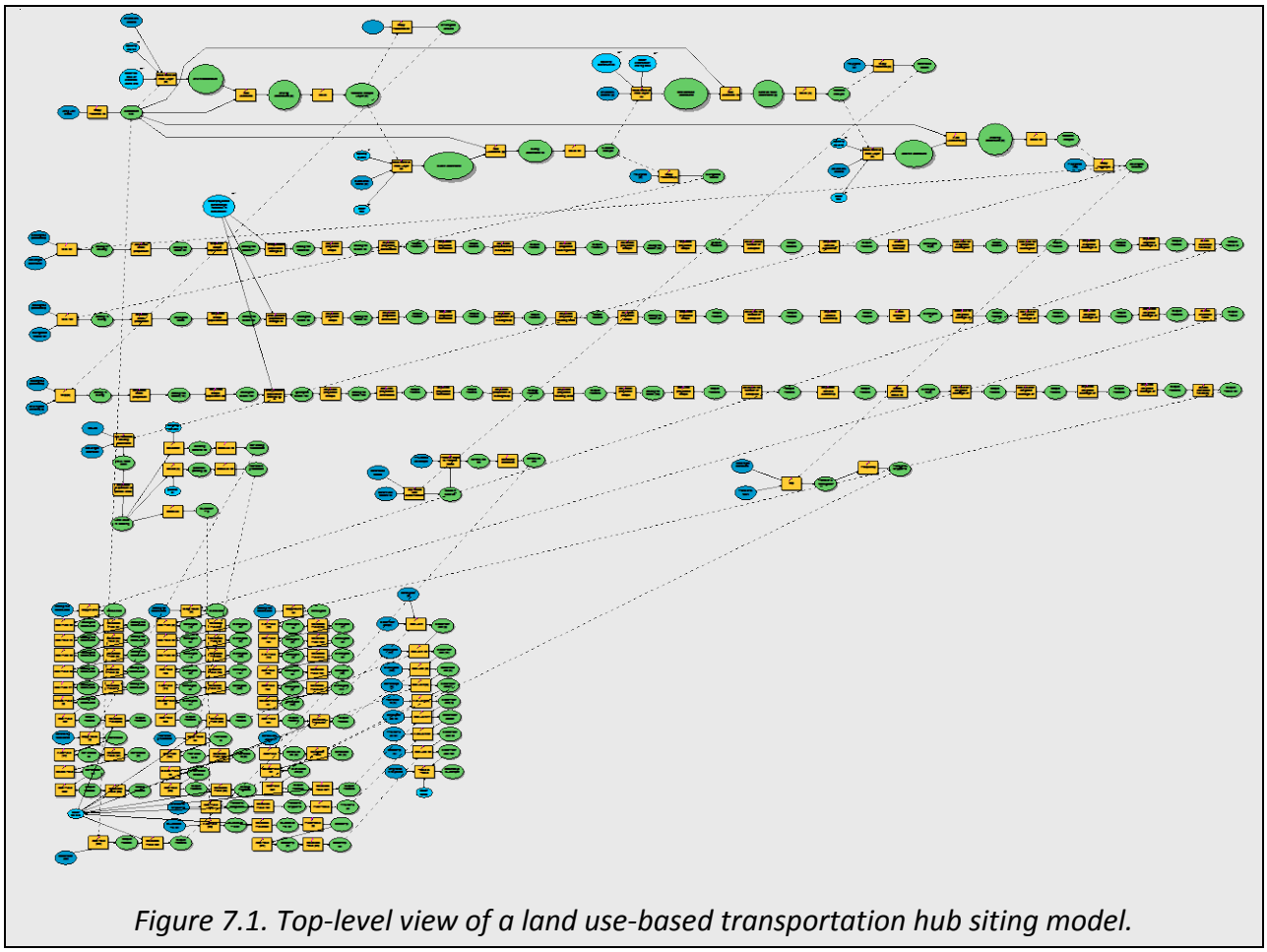

Figure 7.1. Top-level view of a land use-based transportation hub siting model.

As case study 7-1 clearly demonstrates, there are situations where the inner working of the system should be hidden from the user. In addition to task analysis, which we have carried out, we should also note the pioneering work of Andrew Frank and Max Egenhofer at the University of Maine in the early 1990s. Volta (1993) separated the formalization of the problem domain – that is identifying the objects a user manipulates, and their pertinent operations - from its visualization, though interaction elements such as windows and dialog boxes. Bruns (1994) tried to bridge the cognitive dissonance between table top and map space by creating so-called direct manipulation interfaces, where the user combines iconic representations of GIS objects aiming at creating a visual map algebra, which is a common process of integrating different layers in a GIS through algebraic transformation. We will revisit the notion of visual representations of analysis in the section on GIS workflows and process models. Jackson (1990) suggested a series of mappings between three domains: the source domain, the target domain, and the user interface domain, or visualization. In his thesis 'Visualization of metaphors for interaction with GIS' he developed formal specification methods to describe the domains involved and identify similarities and dissimilarities between them. Golledge's (1995) approach

to the same dissonance problem was to identify spatial primitives as building blocks of higher-level GIS operations and subsequently tasks. Albert and Golledge's (1999) thorough investigation of the cognitive aspects of map overlay and Riedemann's (2005) parallel work on topological operators seem to have brought a closure to the topic, to some extent.

7.3. Formalized analysis of GIS user interfaces

With the advent of GIS as a potentially interesting research object in computer science, more formalized approaches to the study of GIS user interfaces began to appear. Kösters *et al.*, (1996a,b) took well-established methods from classic object-oriented systems analysis (OOA) and applied in parallel to (what developers believed are) the domain of GIS tasks and to the analysis of the user interface itself. Use of the same tool was supposed to guarantee that the 'mapping' referred to above could take place. If was a formidable effort that eventually failed because of the complexity of desktop GIS, especially since they have tried to tackle a complex traffic simulation system, and the subsequent size of the formal analysis model. Li and Coleman (2005) use a similar instrument (now based on the unified modeling language) to describe workflows, not on the level of individual GIS operations but at that of user tasks. Their goal is quality control and the formalized approach certainly goes a long way assuring that.

Oliveira and Medeiros (1999, 2000) use considerable effort to formalize the static components of the user interface as well as a procedural view of how these are utilized. While their study objects are GIS and spatial decision support systems, this line of research of a group of information scientist who we will encounter in other parts of this chapter is decidedly not about user tasks or functional GIS components. The result is a dissection into software components (as in COM or CORBA modules) that are great from a software engineering perspective but are entirely void of any user experience considerations.

7.4. User experience considerations

Given the steep learning curve of GIS software, the user interface needs to be adaptable, i.e., work for the novice user as well as for the expert one. According to Mark (1992, p. 551), "a main objective of GIS is to allow the user of the system to interact vicariously with actual or possible phenomena of the world. If this is so, then the system which mediates between the user and the world should be as unobtrusive as possible". He goes on to distinguish between kinds of users (geographic knowledge), their computing skills, frequency of software use, cultural and personal background, and only then the purpose of the user's application. Davies and Medyckyj-Scott (1996) studied the first half of this list and concluded what seems obvious but had not been proven before: experience matters. Here, experience is the sum of educational level, familiarity with the software, and most significantly the match (or lack of) between the developer's and the user's conceptual model of the task at hand.

A user's experience is obviously strongly influenced by the culture she grows up in and the language they speak. Mike Gould (1989) built his early career on exploring the different spatial metaphors in English vs. Spanish and the effect that his on GIS usage. David Mark (1992) extended this to look at a range of other languages such as French, German, or Croatian. Following the work of Lakoff (1980) and Norman (1990), spatial metaphors and image schemata became a fertile research topic (Gould and McGranaghan 1990; Kuhn and Frank 1991; Mark 1992; Nyerges 1992).

Of particular interest is Nyerges' contribution, as he tries to identify cognitive primitives that can be used as building blocks representing elementary spatial relationships such as container, part-whole, linear order, link, center-periphery, etc. we will revisit this theme in section 7.6. Bunch and Lloyd (1996) give credence to the fact that managing the cognitive load experienced by learners is the key to representing geographic information.

Promising as this work looked back then, it has since only made advances in linguistics and cognitive science but had no lasting effect on the design of GIS applications. As to cultural differences mentioned by Mark above, the notion that culture matters was picked up by social critics of GIS (Pickles 1995; Rundstrom 1995), but in the long run, market forces have proven to be stronger: GIS applications still look very much the same, no matter whether they are used in a London finance office, a school on a Navaho reservation, or a Nepalese erosion prevention program.

7.4.1. User requirement gathering in the age of Web Mapping 2.0

Research on the material presented in the previous sections has not been very active since the mid-1990s. However, because of the transition of GIS applications from traditional desktop environment to distributed web services, there is a renewed interest in descriptions of the system and bridging the cognitive gap with the user. Traditional application development methodologies such as Joint Product Design, Joint Application Design, Rapid Application Development, and more recently agile methods, assume that we can foresee the type of geographical questions to be asked and the characteristics of the application's user. More important, textbooks (Downton 1991; Whitten and Bentley 2005) advise us that we work with the user in reviewing the requirement list. However, in the new range of application that emerged with the growth of Web 2.0 environment, these parameters are not available anymore (Kazman and Chen 2009). Baranauskas *et al*. (2005) tried to address this problem by moving onto the web and conducting a public participation process to solicit input from potential users of a web-based GIS. Subsequently, and non-withstanding Brewer's (2002) call-

to-arms, the only recent user requirements gathering for spatial applications are about mobile devices (Queiroz and Ferreira 2009).

7.5. Task analysis as the basis for workflow management

With the user as a criterion out of the picture, we need to refocus on the set of functionalities available in GIS and related software. In the 1980s and 90s, Goodchild (1988), Lanter (1994), Jung and Albrecht (1997), and probably many more compiled lists of GIS functionality. In the age of Google's spatial products, the incorporation of spatial data structures and functions in the statistics packages, and numerous open source tools with varying degree of analytical functionality, the notion of what constitutes a GIS is a bit more fuzzy. The challenge is to match functionalities with generic user tasks, and most importantly to find means of communication that allow for that match to happen not just in theoretically but in reality, during a product design process. Albrecht (1998) did this for desktop GIS, and Klien *et al*. (2006) updated this line of work for web-based applications in disaster management. The following is a review of task analysis for geospatial applications.

Still on the desktop side, Oliveira *et al*., (1997) developed a processing environment, which is both more generic that Albrecht's (1998) VGIS framework and because of that less directly applicable to a GIS implementation. As such, it serves more as an application programming interface (API) for rapid prototyping of workflows in a multitude of application areas that requires a lot of fine-tuning to result in a user friendly application. The advantage of such an approach is the possibility of linking applications from very different domains, though there is virtually no use of standards that we expect nowadays from interoperable software. Originating from the same research unit, Weske *et al*. (1998) describe possibly the same system, now called WASA, which becomes the foundation for two diverging developments. On one hand, they describe something akin to the scientific workflow environments that will be discussed in section 7.6. On the other, WASA is a precursor to a distributed, web-based toolset that is now at the heart of application developments world-wide covered in section 7.7.

Workflow management has become a major focus of business-related information system research. It is a suite of applications and procedures that assist in managing internal workflows inside a corporation and deal with high level tasks, while allowing the monitoring and management of their progress. There are obvious gains in the rationalization of workflows, which have yet not been implemented on any significant scale with GIS packages. One aspect that links with past research is the issue of quality management. Li and Coleman (2005), for instance, conduct their task analysis with the goal of focusing on quality control aspects of work flow management.

Geographers have maintained for generations that (spatial) context matters. Next to location and scale, context is at the core of geographic thought and theory (Spedding 1997, Massey 1999, Gertler 2003) and so it is a little surprising that this theme is receiving more attention within computer science rather than in GIScience. Cai (2007) was the first to formally acknowledge the role of context in the specification of GIS tasks. Cai article ties together several strands of research described in different sections of this chapter. For one, he picks up Volta's formalization of mappings from the developer's view to the user's view, albeit now with modern ontology tools. Cai also employs user input to fill gaps in the specification of context that are to be expected in a distributed Web Mapping 2.0 environment. As such, he manages to keep his ontologies vague, with the system adding constraints on the fly as the user assembles the application. Liu *et al*. (2007) is a first report by a new research group at the National Center for Supercomputing Applications (NCSA) that picks up the context theme and a range of tools and software environments that help to build task ontologies (in particular for seismic disaster management) and end user ready decision support systems.

Case study 7.2 NASA's Earth Science Gateway

Bambacus et al. (2007) introduce the Earth Science Gateway (ESG), a geospatial web portal designed to support prototyping applications and to reuse data by sharing Earth observations, Earth system modeling, and decision support tools. ESG's interoperability provides easy integration of systems and components through open interfaces for rapid prototyping. In their article, Bambacus et al. (2007) report on a rapid prototyping session during a GEOSS demo in May 2006. As do other countries and agencies, NASA and NOAA collaborate on modeling and predicting Earth science phenomena, such as global atmosphere circulation and wind speed. After the simulation results are produced, they are put into services that comply with the Open Geospatial Consortium (OGC) Web Coverage Service (WCS) and Web Mapping Service (WMS) and the services are registered into different catalogs, such as the Earth Science Gateway. In their workshop session, the participants used the ESG to rapidly prototype an application to identify locations for building wind farms to produce electricity in Hainan province, China.
The workflow consists of three main components:
1) Search ESG using "wind" to find services of interest from tens of wind-related services, such as G5FCST Wind Shear.
2) Add the G5FCST Wind Shear service that was found to the viewer to get the desired application.
3) The application shows some wind situations but no detailed wind speed information. Other services, such as the Mean Wind Speed service with rough and fine resolution are then added.

This application can then be saved, and whenever a user brings up the application, updated observations and simulations will be integrated.

Through this process, an application can be prototyped quickly with the support of the ESG. In the process, users do not have to know who provided the observations or simulations, or who provided the external WMS. Professional users only focus on the application logic by searching available services and selecting needed services. Public users only need to bring up the application through an Internet hyperlink prepared by a professional. In this example, the legacy system of collected observation data and simulations of wind speed are leveraged in the find and bind process.

The ESG has the capability to bring up 3-D and 4-D visualizations and is targeted to serve the communities in sharing global earth observation data and simulations. Therefore, the ESG better serves the purpose of rapidly prototyping national applications and GEOSS applications than other more generic portals.

Services found through the ESG discovery functions are limited by the availability and the quality of the service, which depends on several factors, such as 1) the accuracy of observed data; 2) the quality of simulation models; 3) the quality of post-processing of information to provide the service; and 4) the reliability of the service.

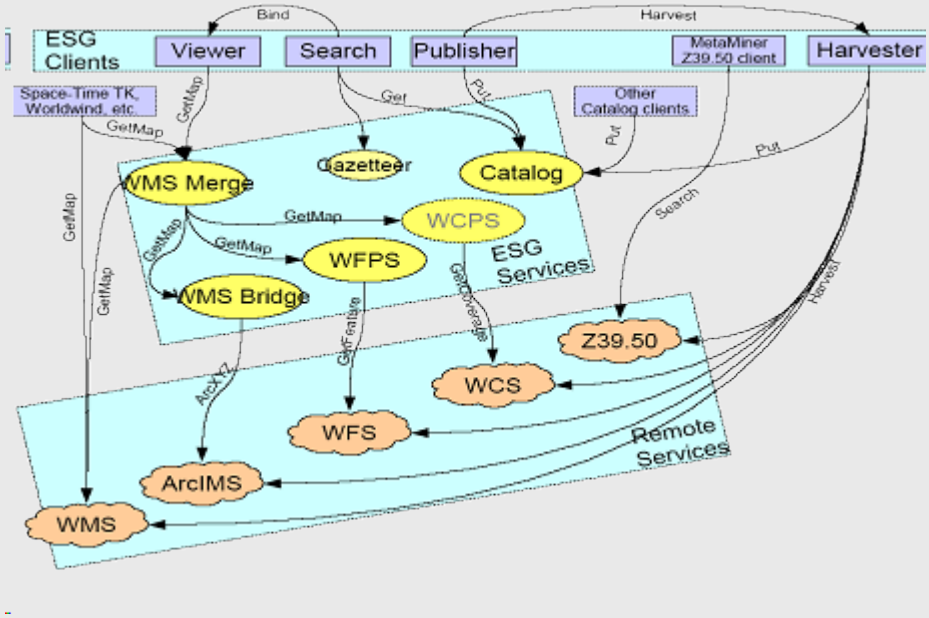

Figure 8.2. Web services chaining for the Earth Sciences Gateway.

In the context of the Alexandria digital library project, a research group around Terry Smith (1995) developed in the mid 1990s a first geoscientific processing workbench. Alonso (1999) continued this work and established with OPERA a programming environment that has become the tool of choice for the next ten years. While Smith and Alonso focus on the concatenation of

functions (not necessarily GIS functions but also shell scripts, Java applets, etc.), others like Bennett (1997), Marr *et al*. (1997), and Jung and Albrecht (1997) focused on the development of process libraries. These are modules that represent physical or social spatial processes, encoded in some standard formalism such as a Stella™ model. Similar to Oliveira and Medeiros (2000), Liu *et al*. (2005) use a component-based approach to the definition of geo workflows. Theirs is a more sophisticated implementation of the kind of tools that Marr and Albrecht used to define their process libraries. The problem with these implementations is that they employ highly customized, non-standard tools that decrease the likelihood of wide-spread use.

7.6. Geo-scientific workflows and process models

To address this problem, a coalition of researchers from the universities of Santa Barbara and San Diego, developed a workflow system that now has the flexibility we were missing from Oliveira's earlier work. Based on a profusion of standards, the Kepler system (http://kepler-project.org) allows for the specification and documentation of a wide range of scientific workflows in distributed environments. Their work is in direct logical continuation of Smith and Alonso; Jäger *et al.* (2005) provide an application of Kepler for geoscientific workflows.

Case study 8.3 Kepler

In scientific workflows, each step often occurs in different software (ArcGIS, R, Matlab, RePast, C#). The Kepler scientific workflow system contains a wide variety of analytical components (e.g., spatial data functions and support for external scripts), allows direct (real-time) access to heterogeneous data, and supports models in many science domains.
Figure 8.3a illustrates some of the system's components using a trivial example. The Director controls the sequence of actor execution. Each actor takes data on its input ports, processes that data, and sends results to its output ports. Actors transform input tokens into output data tokens which then get passed to the next actor under control of the director (Figure 8.3b). Each workflow component is self-documenting as it is defined; the specification becomes automatically part of an ever growing component repository (http://library.kepler-project.org/kepler/ and Figure 8.3c). For scientists, this becomes a boon because such models can not only be easily shared but also referenced in published papers (Figure 8.3d).
Kepler uses hierarchies to encapsulate complexity (compare this with case study 8.1). Composing models using hierarchy promotes the development of re-usable components that can be shared with other scientists.

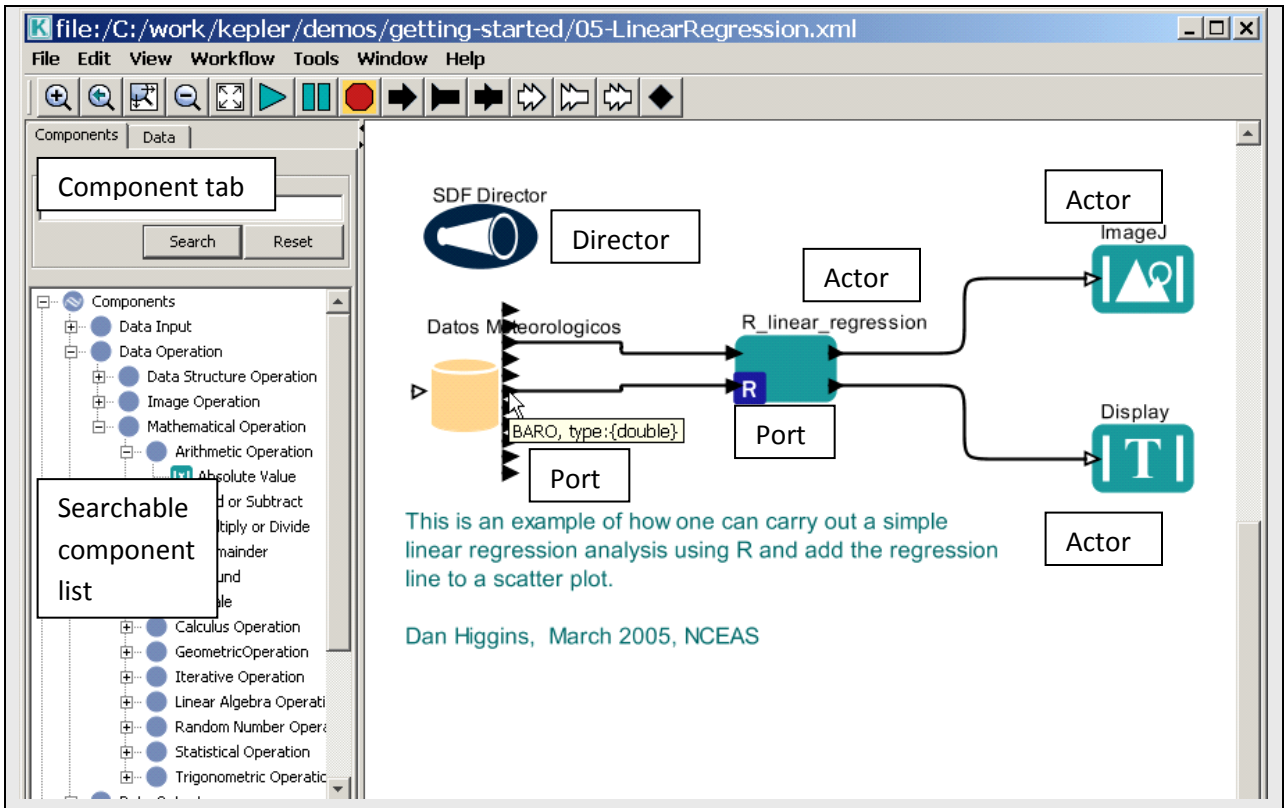

Figure 7.3a. Important components of the Kepler user interface

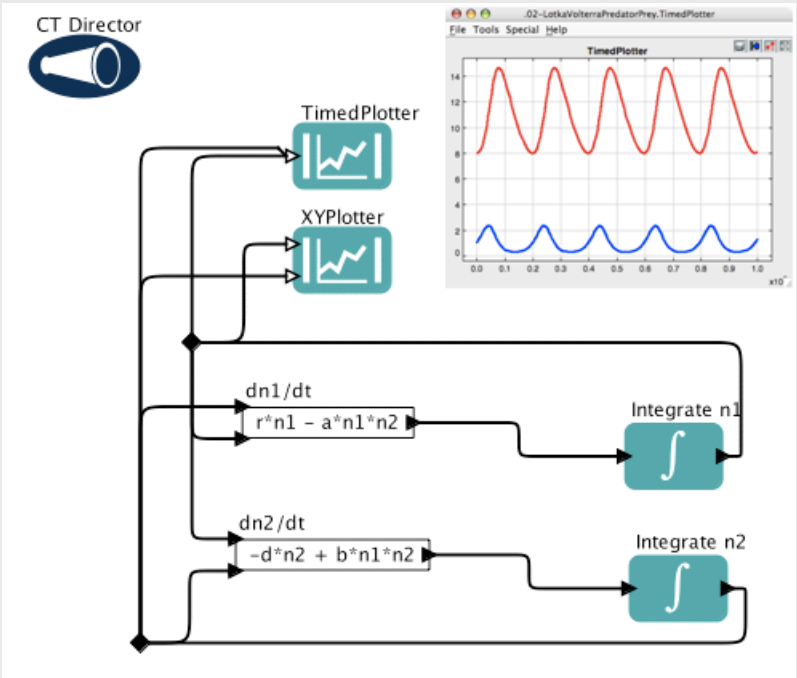

Figure 7.3b. Kepler execution is similar to Stella™.

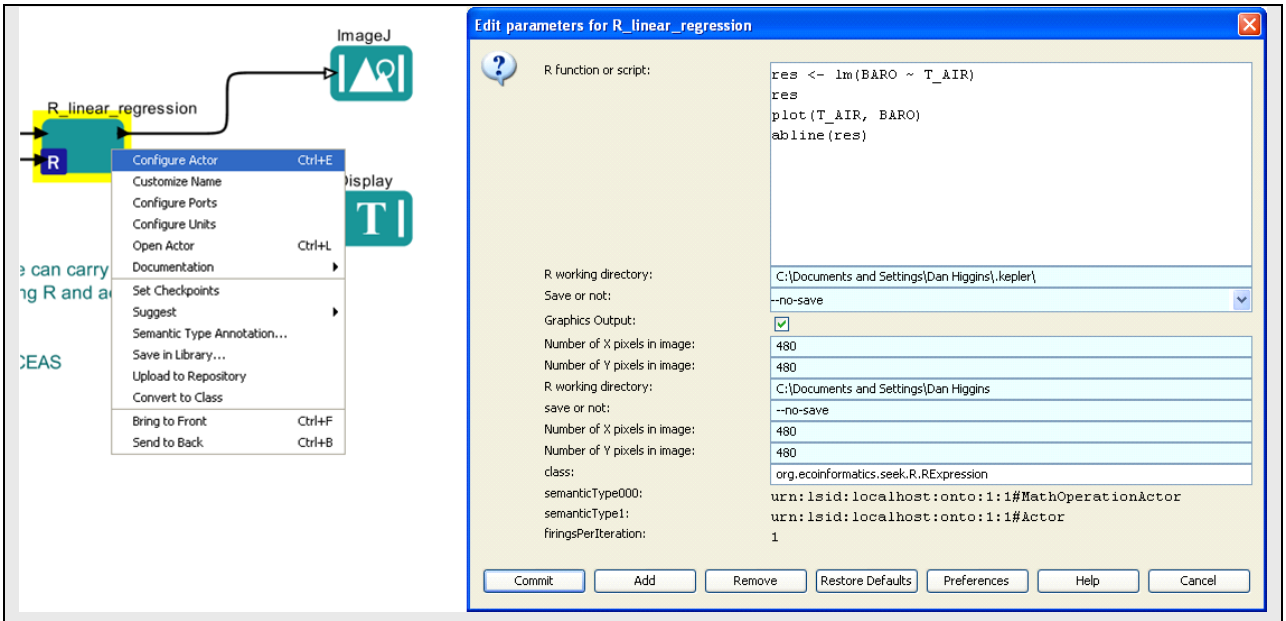

Figure 7.3c. Configuring a Kepler component may involve setting a number of parameters

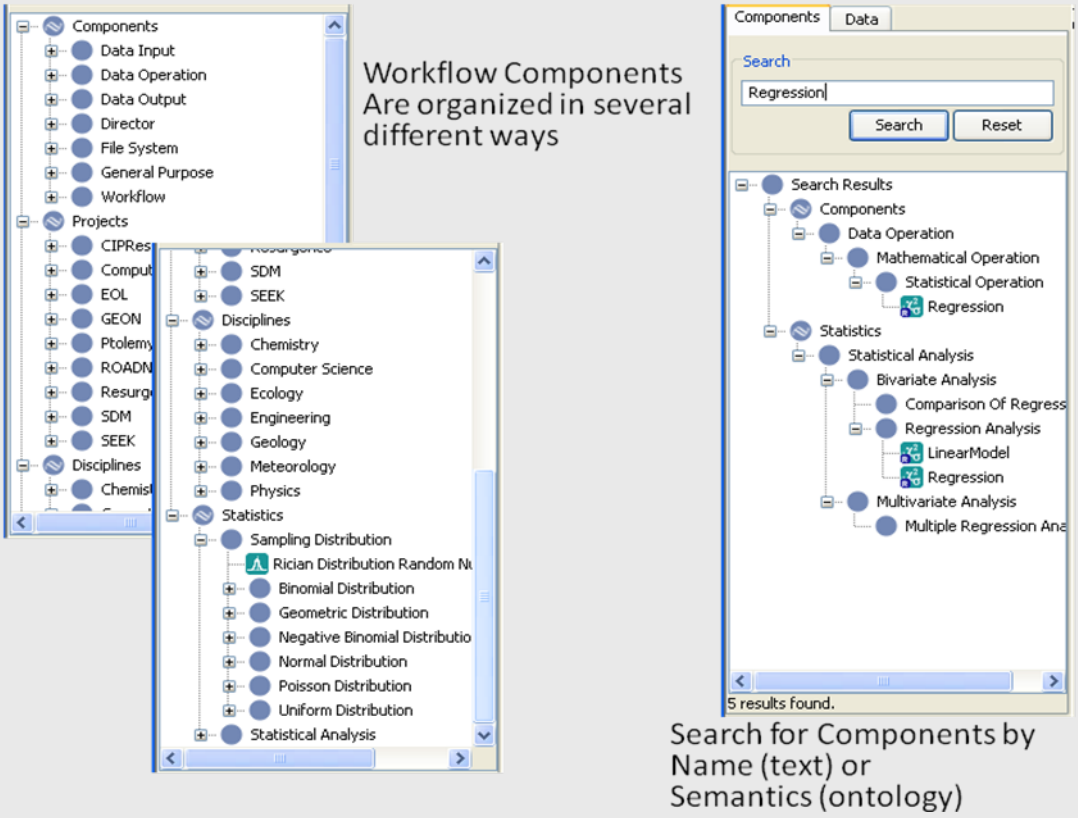

Figure 7.3d. Searching the Kepler component repository.

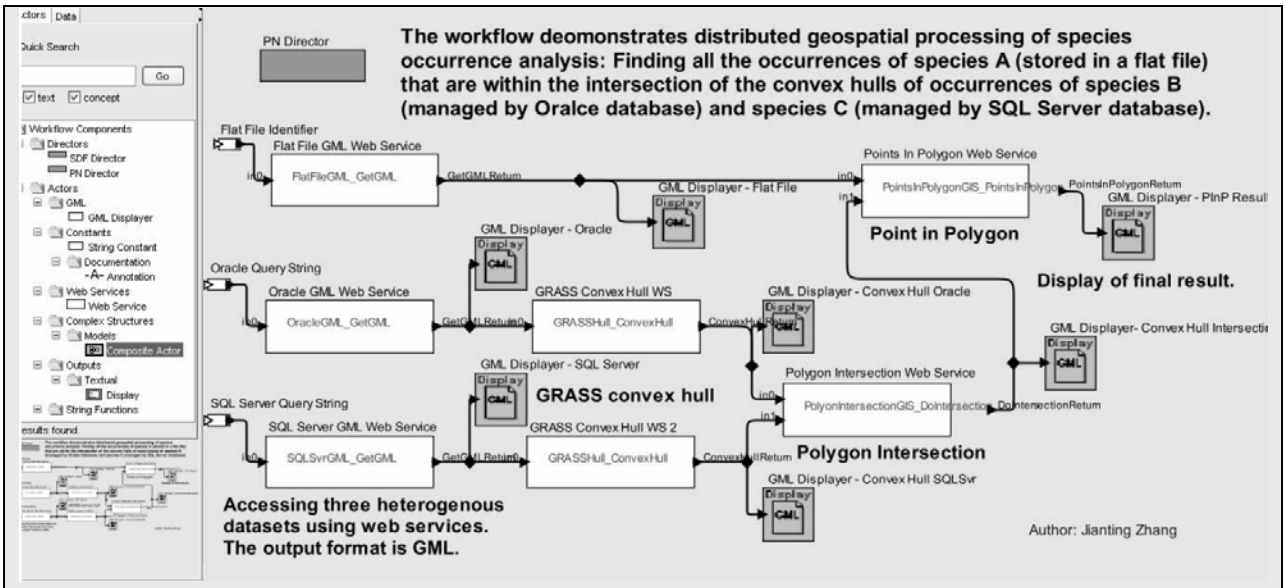

Figure 7.3e. Sample Kepler workflow modeling species occurrences. (from Jäger et al., (2005), page 90).

7.7. Ontologies in support of application planning for the semantic web

One of the characteristics of the Kepler system is its adaption to large scale distributed computing (Grid computing). Complementary to the work described in section 7.4.1, most academic GIS applications move now into an open source-based, distributed processing realm. The key to the success of such endeavors is the use of computer based ontologies that allow formalizing conceptual models of spatial data and processes, which in turn are a pre-requisite to the mapping from one domain to another (Timpf 2001). At a fairly basic level, using a simplistic use case from a homeland security working group, Wiegand and Garcia (2007) develop a task-based ontology to search and combine data from different sources.

Medeiros' group, too, has moved towards web-based user interfaces. We already mentioned Baranauskas *et al*. (2005) attempt to solicit input from potential users of a web-based GIS. Schmiguel *et al*. (2004) contributed the analysis of user interfaces for web-GIS and Medeiros *et al*. (2005) describe a full-fledged workflow-based spatial decision support system in a web environment.

As mentioned before, ontologies are seen as key to the discovery and the combination of web-based functionality. Lutz (2007) describes the use of the Web Ontology Language variant OWL-S to formalize web services that can then be chained into workflows. His article illustrates how cumbersome it still is to develop user interface service descriptions, overcome the inflexibility of reasoning implementations, and map between domain ontologies. At the same time, he shows the way to develop robust web-based GIS applications without having to resort to

Kepler, which in the end is still a tool for the academic community and thus assumes high levels of technical and domain knowledge rather than for end users.

7.8. Summary

GIS application planning has changed dramatically since the early 1990s. Where traditional work involved cognition and reduction of user interface complexity, the emphasis is now on ontological tools that help to bridge between the inherent complexity of the software and the task-oriented conceptual model of the user. However, for many applications the core demands for bridging the cognitive gap between users and internal representation are still key. The same tools are also used to modularize GIS functionality to repackage it as workflows in a web-based environment. The use of advance workflow and ontological research might change the face of geotechnologies over the coming years, though most implementations are still focused on research rather than practice.

7.9. Further reading

Cai (2007) provides breadth of discussion into the issue of application planning and Lutz (2007) for its methodology and level of detail. Davies (1998) provides a detailed discussion of task analysis with the differentiation between work and enabling task, including an analysis of GIS users' tasks. For a detailed discussion of task analysis in general, see Sharp et al. (2007). The very readable PhD thesis of Lemmens (2006) provides a detailed explanation and information. You may also want to keep an eye on the continued development of Kepler and its European counterpart Taverna. Finally, bookmark the web site of Claudia Medeiros (www.ic.unicamp.br/~cmbm/); her group has developed by far the largest body of work on HCI and GIS in the area of the semantic web and application development.

Revision questions

1. In a distributed web environment, what are the tasks that are common to all applications?
2. Does the phrase 'spatial is special' still make sense when we look beyond desktop GIS?
3. What is the role of the user in GIS application planning? Given the limitations outlined in this chapter, how can she be accommodated for?

4. What is the most important skill set for (a) a developer, (b) and academic interest in 21st century application planning?
5. How do mobile device change the context for GIS application development? For a first glimpse, browse the spatial application on the iPhone™ apps store. What works, what doesn't? Can you identify major gaps in the provision of applications on this particular hardware platform?