Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
ISPRS Journal of Photogrammetry & Remote Sensing 60 (2005) 34 – 47
PHOTOGRAMMETRY & REMOTE SENSING
www.elsevier.com/locate/isprsjprs

IPODLAS—A software architecture for coupling temporal simulation systems, VR, and GIS

Daniel Isenegger [a,*], Bronwyn Price [b], Yi Wu [c], Andreas Fischlin [b], Urs Frei [c], Robert Weibel [a], Britta Allgöwer [a]

[a] *GIS Division, Department of Geography, University of Zurich, Winterthurerstr. 190, 8057 CH—Zurich, Switzerland*
[b] *Institute of Terrestrial Ecology (Terrestrial Systems Ecology), ETH Zurich (ITÖ), Switzerland*
[c] *Remote Sensing Laboratories, Department of Geography, University of Zurich, Switzerland*

Received 6 October 2005; received in revised form 10 October 2005; accepted 11 October 2005
Available online 15 November 2005

Abstract

Environmental processes often vary in space and time and act over several scales. Current software applications dealing with aspects of these processes emphasize properties specific to their domain and tend to neglect other issues. For example, GIS prefers a static view and generally lacks the representation of dynamics, temporal simulation systems emphasize the temporal component but ignore space to a great extent, and virtual reality tends to "forget" the underlying data and models. In order to remedy this situation we present an approach that aims to bring together the three domains; temporal simulation systems, GIS, and virtual reality, and to foster the integration of particular functionalities. This paper concentrates on concepts and requirements for the development of a suitable software architecture using case studies and use cases seen from a GIS-based perspective.

Keywords: GIS; design; integration; simulation; visualization; interoperability; software

1. Introduction and motivation

Alpine landscapes are constantly changing, not only in time but also over space. The understanding of spatiotemporal processes and their interrelations is central to the understanding of the complex behavior of real world systems (Pang and Shi, 2002). Relevant processes might span over several temporal and spatial scales. Therefore, tools for modeling, analyzing, and visualizing such processes should also be able to operate on diverse spatial and temporal scales. What kind of tools should be considered to meet these requirements? *Geographic Information Systems* (GIS) provide powerful functionality for spatial analysis, data integration and storage (Nyerges, 1993) and *Virtual Reality* (VR) systems offer interactive virtual fly-through facilities with highly photo-realistic content (Duchaineau et al., 1997; Meyer et al., 2001). These spatially oriented systems lack the ability to represent temporal dynamics and their concepts of landscape are static (Peuquet and Niu, 1995). GIS are very large systems tending to be monolithic, and therefore costly to combine with other systems (Preston et al., 2003). On the other hand, *temporal simulation systems* (TSS) support the simulation of static and in particular dynamic dependencies. Due to the hierarchical structure of state-of-the-art simulation

* Corresponding author. Tel.: +41 1 635 52 57; fax: +41 1 635 68 48.
E-mail address: disen@geo.unizh.ch (D. Isenegger).

0924-2716/$ - see front matter © 2005 International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS). Published by Elsevier B.V.

doi:10.1016/j.isprsjprs.2005.10.003

models, the composition of complex systems through the coupling of models is possible. A drawback, however, is that in general the spatial dimension is neglected or only poorly represented (Fedra, 1993).

1.1. 'IPODLAS'—coupling TSS, GIS, and VR

The goal of this project is to combine the paradigms and concepts of the three domains TSS, GIS, and VR and to exploit their particular strengths to improve the representation of spatiotemporal and cross-scale processes taking place in the landscape. To this end we are developing the system IPODLAS: interactive, process oriented, dynamic landscape analysis and simulation. The IPODLAS project develops concepts for the information exchange between the different types of subsystems (the TSS, the GIS, and the VR subsystem). Thus an integration of the functionalities of the different subsystems can be achieved. A primary goal is to derive concepts and interfaces for a system that is able to model, analyze, and visualize spatiotemporal and cross-scale processes. The focus is to determine the characteristics and functionalities a system like IPODLAS must include. Three case studies provide realistic data and models supporting the development of IPODLAS. To cover the broad range of possible requirements within landscape analysis, representative case studies have been chosen from very different realms, such as insect population dynamics, wildland fire modeling, and human infrastructure modeling.

1.2. Objectives

This paper focuses on the workflow from a user's requirement through to concepts specifying the architecture of a software system. The workflow comprises of a collection of functional requirements of users within use cases (cf. Section 3.2.1), functionality listings, the design of a software architecture supporting these requirements and implementation of a prototype. Aside from software architecture design, we also aim to present the derivation of requirements and concepts that are important for the design of a system with the requirements of IPODLAS in an exemplary manner.

2. Issues of combination of GIS, VR, and TSS

Taking into account the, in some aspects, complementary strengths and weaknesses of the respective subsystems it seems to be a promising approach to combine them into a common framework. By combining concepts and paradigms of the three domains and exploiting their particular strengths, the investigating of spatiotemporal and cross-scale landscape processes forming the landscape can be improved. The benefits of combination and potential synergies of GIS and TSS (Bernhard and Krueger, 2000; Brimicombe, 2003; Fedra, 1993, 1996; Goodchild et al., 1996; Raper and Livingstone, 1995; Vckovski, 1998) as well as of GIS and VR (Camara et al., 1998; Huang et al., 2001; Lindstrom et al., 1997; Pajarola and Widmayer, 2001) are widely acknowledged. Combining the 'trio' GIS, VR, and TSS promises gains through cross-fertilization and mutual support, but it is conceptually and technologically complex. One underlying core problem is the differing data models used in GIS and TSS (Aspinall and Pearson, 2000; Bennett, 1997; Fedra, 1993, 1996). In the GIS, the data model is centered on representations of the geographical space, the objects located there and their relationships to each other. The focus is on location and topology. TSS data models are designed to model processes, their states, and throughputs of quantities. GIS is designed to model static representations, whereas TSS specializes in model dynamics (Fedra, 1996). These differing emphases result almost necessarily in different conceptual and technological structures (Brimicombe, 2003).

2.1. GIS functionality used

For TSS, the principal benefit of being linked to a GIS is gaining the ability to deal with large volumes of spatially oriented data. Major environmental tasks such as inventory, assessment, management, and prediction in diverse research areas such as atmospheric modeling, land surface–subsurface modeling, and ecological systems modeling can be supported with GIS functionality. The primary modes of GIS usage in practical applications are map, query, and model. The map mode offers to browse information using standard methods such as pan and zoom. In the query mode, the user queries information about locations and/or phenomena. Finally, in the model mode, GIS supports model usage. Although the usage of all primary GIS functionality data entry/capture, data storage/management, data manipulation/analysis, and data display/output can be beneficial to a TSS (Nyerges, 1993), GIS is frequently used only as a pre-processor to prepare spatially distributed input data and as a post-processor to display and analyze model results (Bennett, 1997; Brimicombe, 2003; Fedra, 1993, 1996).

2.2. Integration strategy

The degree to which different systems should be coupled has been and still remains a subject of

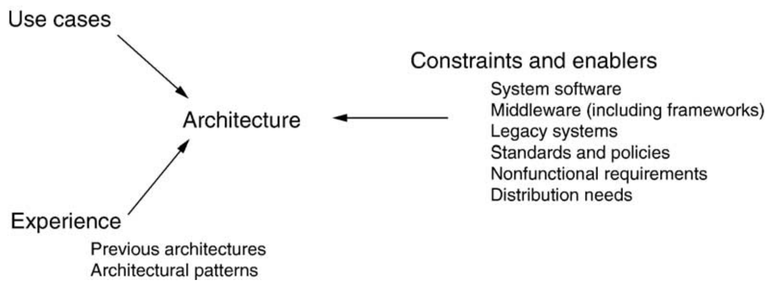

Fig. 1. Requirements and constraints influencing the architecture (Jacobson et al., 1999).

investigation (Brandmeyer and Karimi, 2000; Brimicombe, 2003; Fedra, 1993, 1996; Nyerges, 1993). Brimicombe (2003) suggests what he calls a 'maturing typology' of GIS and environmental model integration ranging from one-way data transfer and loose coupling over shared and joined coupling to tool coupling. The appropriate integration strategy depends on the properties and aims of the respective projects. There are always tradeoffs between contradictory goals, e.g., between efficiency and the flexibility of a system or between the ease of use and the costs of development (Fedra, 1996).

2.3. Interoperability initiatives

In the GIS domain, the OpenGIS Consortium (OGC) (OGC, 2005) defines platform independent, generic interfaces making up a framework supporting interoperability for GIS components (Bernhard and Krueger, 2000; Buehler and McKee, 1998), but it fails to define explicitly the representation of temporal aspects (Schulze et al., 2002). In contrast in the TSS domain, the High Level Architecture (HLA) (IEEE, 2000) provides a framework for distributed time-variant simulation processes. HLA is a federation approach and focuses on interoperability and reuse of simulation. HLA however lacks the support of spatial applications, hence its shortcomings can be considered complementary to the ones of the OGC standards. Simulation models based on the theory of modeling and simulation, discrete event systems specification, and knowledge-based simulation methodologies (De Vasconcelos et al., 2002; Zeigler, 1976, 1990) support due to their hierarchical structure the composition of complex systems through the nesting of models. Although both interoperability approaches remain limited to their respective domains, OGC and HLA provide a promising foundation upon which the integration of the domains GIS and TSS might be built (Schulze et al., 2002).

The Institute of Electrical and Electronics Engineers (IEEE) defines interoperability as the ability of two or more components to exchange information and to use the exchanged information (IEEE, 1990). Emerging concepts from the IT domain provide possibilities to deal with the technical issues associated with integration of different systems. These are, for example, layered architectures used in distributed computing (Ghezzi et al., 2003), the idea of web service trading (Reference Model for Open Distributed Processing, ISO/IEC 10740) (Schulze et al., 2002), or XML-based languages. The application of the client–server paradigm to a Web Map Service or Web Feature Service is an example of 2-tiered architecture (Ghezzi et al., 2003). The addition of a mediating layer is fundamental for 3-tiered architectures such as CORBA[1] (OMG, 1999). The coupling of different GIS or DBMS (Bergmann et al., 2000a,b; Preston et al., 2003) are integration examples from within the GIS domain, the mediation-based framework of Yates and Bishop (1997) and the layered architecture of Bernhard and Krueger (2000) integrates modeling systems and GIS. XML-based languages are used to capture not only formats but also to describe semantics of the information exchanged (Bergmann et al., 2000b; Preston et al., 2003).

3. Methods and materials

3.1. Unified software development process

The ***Unified Software Development Process*** (UP) (Jacobson et al., 1999) has been used to develop IPODLAS. The goal of the UP is to transform user requirements into a software system. As Fig. 1 shows, *Use cases* (cf. Section 3.2.1) are applied to determine the

[1] Common Object Request Broker Architecture (http://www.corba.org/).

functional requirements. *Constraints and enablers* summarize conditions which must be taken into account when designing a software system (Jacobson et al., 1999).

3.2. Constraints and enablers

3.2.1. Case studies and use cases

Case studies provide real-world data from the test area(s) and simulation models supporting the development of IPODLAS. They help to reduce complexity and act as a test bed for the concepts and applications being developed. Case studies also help in communicating results to potential end-users.[2] A use case specifies a concrete scenario from a case study. All use cases together jointly make up the use case model, which may cover the complete functionality of the planned system. A user definition specifies the general intentions of the user which influences his or her requirements for a software system (Jacobson et al., 1999).

3.2.2. Legacy systems

The applications listed below were chosen because they provide functionality typical of applications in their particular domains and required by IPODLAS to satisfy the requirements captured in the use cases and defined in the functionality lists. RAMSES (Research Aids for Modelling and Simulation of Environmental Systems)[3] has been evaluated to be one of the most appropriate TSS for the needs of IPODLAS (Giorgetta, 2002). RAMSES supports modeling and interactive solving of non-linear differential equations, difference equations, and discrete event systems in any combination (Fischlin, 1991). GRASS GIS (Geographic Resources Analysis Support System)[4] is an open source GIS with raster, topological vector, image processing, and graphics production functionality that operates on various platforms (Neteler and Mitasova, 2002). The subsystem chosen to represent the virtual reality domain is VTP (Virtual Terrain Project).[5] Its goal is to facilitate the creation of tools for interactive, 3D visualization of the earth by bringing together the domains of GIS, visual simulation, surveying and remote sensing.

3.2.3. Standards, policies, and languages

To support the interchangeability of the subsystems, we divide the whole system into subsystems, that is, the design is modular. The modules are determined by high cohesion within the module and low coupling between the modules (Ghezzi et al., 2003). They expose their interfaces and hide their implementation (Preston et al., 2003). The resulting reduction of the communication load between subsystems limits the dependencies between the individual subsystems and therefore supports their interchangeability. Applying interoperability standards to the design of the interfaces using standard communication protocols facilitates compliance of other standard components and therefore the interchangeability of components, augmenting the stability of the interfaces. An example for this is the use of a language of the XML family (XML, 2004) and the application of internet sockets (Stevens, 1990).

The Geography Markup Language GML (Lake et al., 2004; OGC, 2003) is an XML (XML, 2004) extension for encoding the modeling, transport, and storage of the spatial and nonspatial properties of geographic features. The key concepts used by GML to model the world are drawn from the OpenGIS Abstract Specification(OGC, 1999) and the ISO 19100 series (ISO/TC, 2004). The use of GML is expected to lead to greater interoperability between applications within the GIS world and to facilitate data sharing (Preston et al., 2003). With the advent of GML 3.0 (OGC, 2003) the use of temporal information and dynamic features is supported, i.e. there are structures to store and transport temporal information (Preston et al., 2003). A major disadvantage of XML-type languages is the inflated data volume due to additional metadata and the use of a text-based format for encoding binary data. In addition, the parsing of the XML data makes the computer performance critical. Compressing the data and transferring binary data separated from the format description can relief this problem to a certain extent (Hoheisel, 2002).

4. Identifying the required functionality—the IPODLAS approach

This paper addresses methodological aspects. The methodology required to develop the concepts and the IPODLAS prototype are seen as results and therefore presented in Section 4. The concrete setup of the case studies and use case applied in the IPODLAS project is described in detail in the following methodological Sections 4.1–4.3. A concrete example of how to get from the prose description of a use case to the

[2] Knowledge based dynamic landscape analysis and simulation for alpine environments. Full Proposal for SNSF Project Nr. 4048-064432 (http://bscw.geo.unizh.ch/pub/bscw.cgi/d77727/nfp48_scient_final.pdf).

[3] http://www.sysecol.ethz.ch/SimSoftware/SimSoftware.html#RAMSES.

[4] http://www.geog.uni-hannover.de/grass/.

[5] http://www.vterrain.org/.

Table 1
Models at three different scales from three case studies

Case study	Simulation model		
	Small scale	Medium scale	Large scale
LBM (larch bud moth)	M8: Local LBM dynamics (no spatial dimension) (Fischlin, 1982)	M9: combining M8 with migration within valley (Fischlin, 1982)	M10: combining M8 with migration between several valleys (Fischlin, 1983; Giorgetta, 2002)
WLF (wildland fire)	*Local Rothermel: describing fire spread in finite elements* (Rothermel, 1972)	SPARKS: combines the Rothermel model with fire spread models covering surface fire (Schöning, 2000)	FARSITE: combines the Rothermel model with fire spread models covering surface and crown fire and fire spotting (Finney, 2004)
MMI (man-made infrastructure)	*Visualizing single buildings in a street in a village*	*Visualizing village and surroundings*	*Visualizing the whole study area*

Models in italics are not yet implemented on the respective subsystem (Allgöwer et al., 2003).

identification of the required functionality of a system is the subject of the Section 4.4. The resulting software architecture which supports the requirements specified in the use cases is characterized in Section 5.

4.1. Case studies

Three case studies from different domains have been chosen to capture a broad range of requirements. The Larch Bud Moth (LBM) case study represents insect population dynamics, the wildland fire modeling (WLF) case study is an example of an abiotic process, and the modeling of man-made infrastructure (MMI) represents a case study where the human impact on landscape is visualized. These case studies were chosen to include both spatial and temporal aspects and to offer models and associated data across several scales (cf. Table 1) (Price et al., 2005).

4.2. Use cases

The use case model consists of a definition of the users and the description of all use cases. In the IPODLAS project, two types of users are defined to cover diverse requirements, the *pilot*-type and the *expert*-type user. The behavior of the pilot user is characterized by

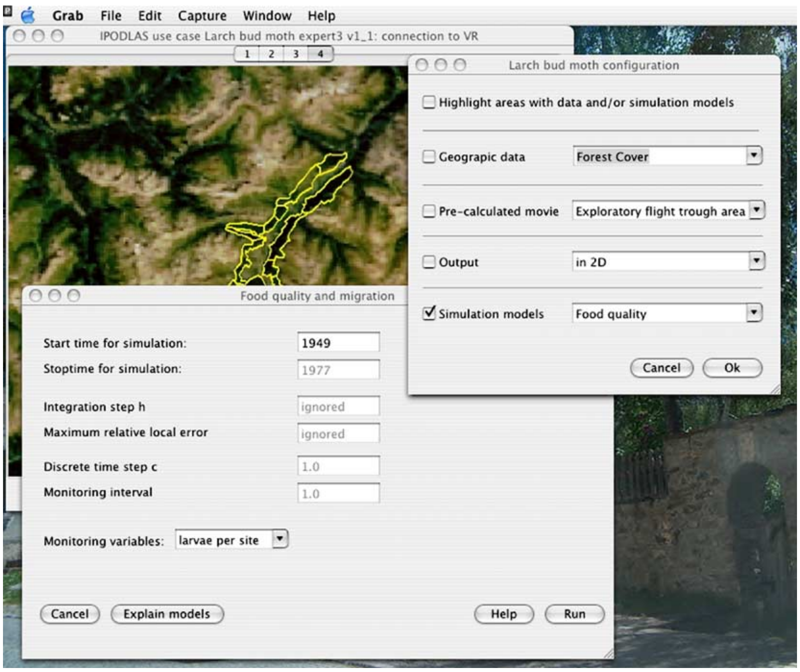

Fig. 2. A screenshot of the IPODLAS GUI defining graphical elements used in an LBM use case.

exploration; she/he flies through the virtual scenery and usually does not change any of the parameter settings but instead uses default configurations when running simulations. In contrast, the expert user is interested in the scientific capabilities of the system; she/he wants to change parameters of the particular subsystems and can plug in new models. Each use case is first described in prose, which defines the particular intentions and the interactions of the user with the system in order to reach the goal of the user described in the related use case. The prose text then is refined in a *sequenced action list*, where the interaction of the user with the system is defined step by step by specifying the input of the user and the response of the system. The graphical definition of the system interface (cf. Fig. 2) helps to specify the state the system is in and the functionality offered.

Among the set of use cases specified, the prospective users select the subset which entails the most important ones. These key use cases may amount only to 5% to 10% of all use cases, but they are the significant ones, as they constitute core system functionality (Jacobson et al., 1999).

4.3. Listing and classifying the required functionality

The use case model consisting of all use cases defines the range of the required functionality that IPODLAS should entail in order to deliver all services the users specify in the various use cases. The functionality recorded in the sequenced action lists forms the basis of the functionality lists describing which functions have to be offered by which subsystem. The functionality list is not only specified from the user's perspective, i.e. seeing the system from outside as a monolith, but conceptually looks to the subsystem level assigning the required functionality to be offered by a particular subsystem. In Table 2, functionality is classified according to estimated implementation efforts of integrating this particular functionality into IPODLAS. The classification of functionality together with the identification of the set of key use cases helps to discover the use cases with the greatest risks of failure.

Table 2
Classification of functionality

Class	Classification of functionality
1.	The required functionality is already implemented in one of the subsystems of IPODLAS.
2.	The required functionality is implemented in another software system.
3.	A solution to offer the required functionality exists in the literature.
4.	An algorithm does not exist in the literature.

4.4. Use case LBM expert 2 (LE2)

Several use cases have been developed in the IPODLAS project, at least one for each user type and for each case study. In the following, a representative example of a use case is described (Price et al., 2005) starting with the definition of the user (cf. Box 1).

Box 1
Use case Bronwyn—the user

> Bronwyn is an 'expert user' of the system IPODLAS. She is a Ph.D. student within the IPODLAS project and wants to use subsystems of the system (GIS, TSS and VR) and the overall system to help her solve research questions regarding the LBM and, then in turn, assist in the development of IPODLAS through provision of data and models and a test bed case study.

The larch bud moth (LBM) population dynamics are spatiotemporal and multi-scaled processes in the Alps. LBM is a forest defoliator causing spectacular damage to larch forests across the Alpine arc, approximately every 9 years (Baltensweiler and Fischlin, 1988). Spatio-temporal dynamics can be modeled by coupling local dynamics models with models of migration between subpopulations at the valley and/or the Alpine Arc scale (Fischlin, 1982, 1983). The prose form of the use case (cf. Box 2) describes the goals of the expert user Bronwyn (Price et al., 2005).

Box 2
Use case LE2—prose description

> Bronwyn is interested particularly in migration of the LBM across the Engadine valley. She wants to see how far LBM migrate per season taking into account wind speed and direction and elements of the landscape which may effect LBM flight such as slope, aspect and local temperature.

Next, the sequenced action list (cf. Table 3) specifies the sequence of interactions of the user with the system (Price et al., 2005).

Table 3
Use case LE2—sequenced action list

Action	Description of action
1.	Bronwyn starts IPODLAS and selects LBM from the list of topics.
2.	IPODLAS shows her the Alpine Arc with highlighted areas, where LBM data can be provided. Bronwyn selects the Upper Engadine valley.
3.	IPODLAS displays a 2D map of the Upper Engadine valley. An additional menu shows several options (geographic data, 3D, simulate, pre-calculated movie). Bronwyn chooses to simulate and see the output in 2D.
4.	Bronwyn chooses a start and stop time (1990, 2000) and otherwise keeps all defaults, then runs the model.
5.	IPODLAS displays a 2D visualisation of the output showing comparative numbers of LBM migrating (departure and landing points).

This use case requires functionality provided from several subsystems. In Table 4, only the actions where the GIS subsystem is involved are specified and classified (in the column 'Class') according to Section 4.3. For the sake of clarity, the actions from the sequenced action list are described in substeps.

The GIS subsystem is only involved in the actions 2, 4, and 5 of the sequenced action list. This can be explained through the division of labor on the subsystem level: storage(s) is accessible by every subsystem and the navigation of the user is handled by the VR subsystem. The functionality listing in Table 4 is typical for the functionality listings of use cases developed for IPODLAS. A major part of the required GIS functionality in the use cases, at least on this level, is standard spatial analysis functionality such as map algebra and map overlay or data integration such as joining attribute data (in textual form) and spatial data. When implementing these use cases the challenges that are occurring at this stage of IPODLAS development is not (yet) missing GIS functionality, but rather the communication between the subsystems. The action numbers 4a, 4d, 5a, and 5c can be classified in functionality classes 2, 3, or 4 depending on the conceptual and technical complexity of the chosen solution to provide this functionality. As an example, for step 4d classification of this task into class 2 could mean that data is sent only as simple text file to the requesting subsystem, while class 3 indicates a more advanced solution such as the automatic encoding of spatiotemporal data in GML3.x for transmission (this will be explained in Section 5.2.). On a conceptual level, in action numbers 4a and 5a, the control flow is affected, i.e. seamless access of functionality is required. Action numbers 4d and 5c address the data flow, i.e. seamless data access. In a study analyzing interoperable and distributed GIS, Bergmann et al. (2000b) present similar findings. The ability of interaction of components through information exchange, in particular seamless data access and access to remote methods is a major requirement to move towards interoperable GIS.

Table 4
Use case LE2—required GIS functionality

Action	Required functionality	Class	Applied standard GIS functionality
2a.	Receiving request to provide areas with available data	2/3/4	Communication/information exchange between subsystems
2b.	Selecting data from storage(s): areas with available data	2	Connection to storage(s) and retrieving data from storage(s)
2c.	Notifying the requesting subsystem about available data via kernel	2/3/4	Communication/information exchange between subsystems
4a.	Receiving request concerning forest and in particular larch distribution, calculating slope and aspect, and wind simulation	2/3/4	Communication/information exchange between subsystems
4b.	Selecting data from storage(s): forest data, larch data, temperature data, DTM	2	Connection to storage(s) and retrieving data from storage(s)
4c.	Calculating: larch per hectare, forest area per hectare, temperature distribution	1	Map algebra, Map overlay, clipping
	Calculating slope, aspect	1	Slope, aspect calculation from DTM
	Simulating Wind speed and direction	2	Querying wind model
4d.	Notifying the requesting subsystem about available data via kernel	2/3/4	Communication/information exchange between subsystems
5a.	Receiving request to transform tabular simulation output to raster	2/3/4	Communication/information exchange between subsystems
5b.	Transform tabular simulation output in raster	1	Join attribute data with spatial data
5c.	Notifying the requesting subsystem about available data via kernel	2/3/4	Communication/information exchange between subsystems

5. Software architecture

The overall goal is to develop an architecture that makes the system resilient to change or change tolerant. The software architecture includes the most important static and dynamic aspects of the design of a system. It focuses on significant structural elements as well as on the interactions that occur among these elements via interfaces (Jacobson et al., 1999). Due to the iterative nature of software development, in some chapters of this paper the individual steps of architecture refinement are split into different phases of development (e.g., early phase, advanced phase). Owing to the modular design of IPODLAS, the dependencies between different architectural aspects are limited and therefore subsystems in different phases should interact smoothly and seamlessly with another. This means that in some architectural aspects the features planned in the advanced phase can be implemented while other aspects remain in the early phase.

5.1. Development of the software architecture

To illustrate the iterative approach, this section outlines some prototypes of IPODLAS which demonstrate the evolution of important parts or phases of the software architecture. The first prototype, the "*Intelligent Tree*", presents a simple interaction model of the subsystems, while the second prototype *Cross-implementation* points at the benefits of the interaction of the subsystems. Section 5.1.3. discusses an example for inter-process communication.

5.1.1. The 'intelligent tree'

The earliest version of the IPODLAS prototype was built to model the growth of an '*intelligent tree*' (Fischlin et al., 2002). The tree was considered as intelligent because it is aware of its location and therefore its growth conditions. In this prototype the user can specify the place where a tree is to grow by clicking with the pointing device within the VR subsystem (cf. Fig. 3). The GIS delivers data related to the habitat conditions (elevation, slope, and aspect) at the chosen location calculated from the DEM, while the TSS subsystem calculates the growth rate according to climate conditions, which are determined by elevation. Tree growth is visualized in a stepwise fashion in the VR subsystem. Data flow is handled by file exchange, while control flow is based on semaphores. Each subsystem is only allowed to access its respective input data file when the semaphore associated with this data file exists. Then the active subsystem has exclusive file access. After termination of all file accessing operations of the active subsystem the respective *Ready-semaphore* R_i associated with the output file of the active subsystem is generated to notify the other waiting subsystems. This initial prototype served as test bed for realizing a concrete division of labor among the subsystems and to test specific communication means such as file-coupling (Fischlin et al., 2002). Exchanging information via files is an acceptable when advantages of simplicity outweigh performance losses. This depends on the operating system's characteristics of generating, reading, writing, and destroying files. A drawback is that synchronization of file access by semaphore files is not very flexible in comparison to a process-based approach with a central coordination process, i.e. synchronization of a more complex system by semaphore is quickly prohibitively challenging.

5.1.2. Cross-implementation

In a subproject known as *Cross-implementation* (Isenegger et al., 2004) a fire simulation model already implemented in the GIS (*Sparks* (Schöning, 2000), cf. Table 1) was implemented in the TSS subsystem and an LBM simulation model already implemented in the TSS

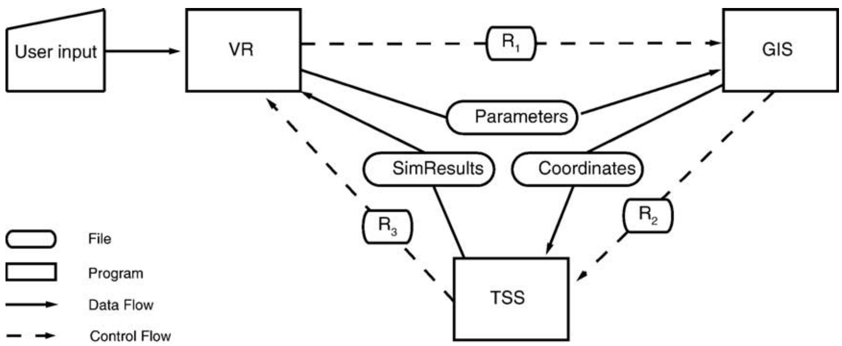

Fig. 3. Overview of the "*Intelligent Tree*" Architecture. R_i *(Ready-semaphore)* represents the respective semaphore files. *Parameters*, *Coordinates*, and *SimResults* are the data exchange files.

subsystem was implemented in GIS (*LBM8* (Fischlin, 1982; Fischlin and Baltensweiler, 1979), cf. Table 1). The aim of this subproject was to discover capabilities and limits of the particular system dealing with problems for which the systems are not designed. That is, the GIS was challenged with a simulation model computing mainly temporal processes and the TSS subsystem with processes with a strong spatial aspect. In this project an ArcInfo 8.1 workstation was used as the GIS subsystem and RAMSES as the TSS subsystem. While RAMSES provides libraries offering sophisticated mathematical and simulating capabilities, it lacks spatial functions, particularly for displaying geo-referenced data, spatial analysis, and storage of large volumes of spatial data. ArcInfo only has limited to no temporal functionality in comparison to RAMSES and besides its performance disadvantages the macro language AML does not support higher programming concepts. Aside from providing the somewhat trivial insight that applications best deal with problems for which they are designed, this project highlighted needed functionality and which subsystem should best provide this functionality (Isenegger et al., 2004).

5.1.3. Socket communication

Fig. 4 illustrates that the TSS subsystem can be controlled over a network applying the client/server approach. A client can trigger the TSS subsystem to start and stop simulation model runs, change the active simulation model and transfer results encoded in XML (Bergamin, 2004). The inter-process communication between the XML-RPC-Server and the TSS subsystem uses AppleEvents, while the communication between the client and the server is done with sockets using XML-RPC protocol (UserLand Software, 2003).

5.2. Current software architecture

As the use cases showed, the interaction of the subsystems follows a certain sequence of (inter)actions requiring some form of synchronization. As a central coordination subsystem, the IPODLAS kernel (cf. Fig. 5) provides this functionality: all control flow is managed by the kernel. The IPODLAS managed storage holds metadata describing the data available to IPODLAS. It is closely linked to the second persistent storage of the IPODLAS system, the GIS storage. All data flow from the storages to the subsystems is managed by direct interaction to avoid the coordination subsystem becoming a bottleneck.

Modularity of the architecture is enhanced by the coordination subsystem, that is, the IPODLAS kernel, limiting the number of interfaces needed for communication between the subsystems. The role of the kernel as the only communication interface between the subsystems means that changes in the interaction of the subsystems or even the exchange of a subsystem, for example the use of another GIS, need only to be registered in the IPODLAS kernel. An exception to the strict modularity and the separation of control and data flow, is the interface between VR and the GUI due to possible heavy communication load between these two subsystems and to real-time requirements.

As reported by Leclercq et al. (1996), each subsystem must map the parts of its data model that are essential for IPODLAS onto the canonical data model of the IPODLAS kernel. To be able to access the

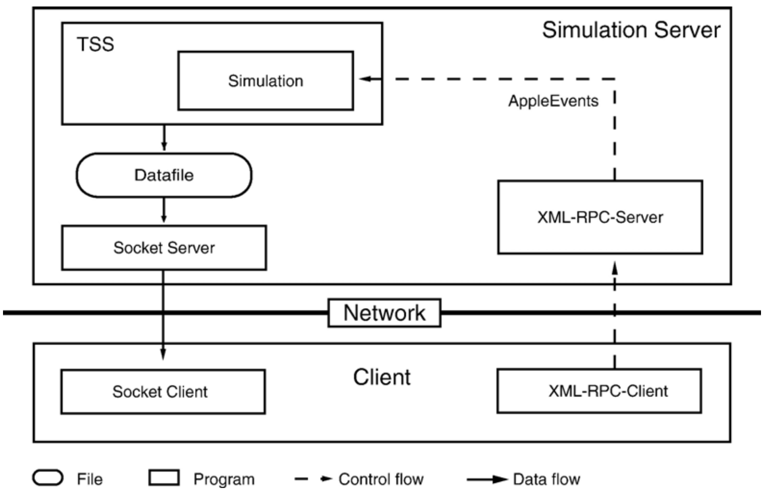

Fig. 4. The socket client/server model of the TSS subsystem (after Bergamin, 2004).

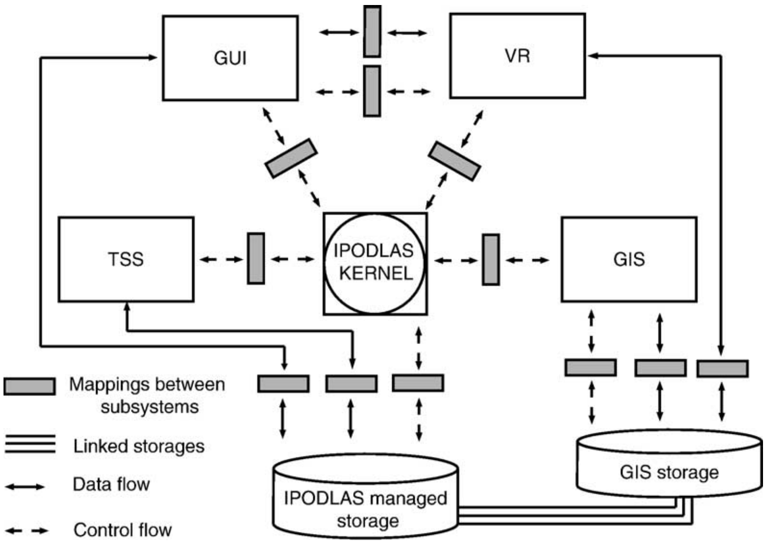

Fig. 5. A schematic view of the IPODLAS software architecture.

functionality of the subsystems functional mapping is also mandatory. In an initial development phase the central coordination process can be kept rather simple, restricted to sequencing the interactions of the subsystems. An advanced solution provides a mediating kernel such as the ones presented in Zaslavsky et al. (2000) and Savary and Zeitouni (2003) receiving requests, dispatching them to the appropriate system, and providing feedback to the user.

5.2.1. Data exchange

In Fig. 6 an example of communication between the GIS subsystem and the TSS subsystem is outlined. The TSS subsystem needs to know the terrain aspect of a certain area for a simulation task and thus sends a request to the kernel (step 1 in Fig. 6), written to a socket connection. The kernel listens to the socket connection, gets the request and dispatches it to the appropriate subsystem, the GIS (step 2). The GIS reads the digital elevation model (DEM) from the storage (step 3) and calculates the terrain aspect values for the DEM (step 4), serializes the result and writes it back to the storage (step 5). Then the requesting subsystem, the TSS, receives the notification through the kernel (step 6 and step 7) of where the aspect data is and reads this information (step 8).

5.2.2. GML 3—temporal aspects

GML 3 specifies the schemas *temporal.xsd* and *dynamicFeatures.xsd* to represent temporal issues. The former schema defines primitives and properties for representing temporal instants and periods. The latter schema allows definition of elements and types to model dynamic features. A *DynamicFeature*, aside from time-invariant properties, entails a *history* property to express the historical development of the feature. The *history* associates the feature with a sequence of time slices which include the dynamic properties of the feature (Lake et al., 2004). Employing those features of GML3 to the LBM case study (cf. Section 4.) augments

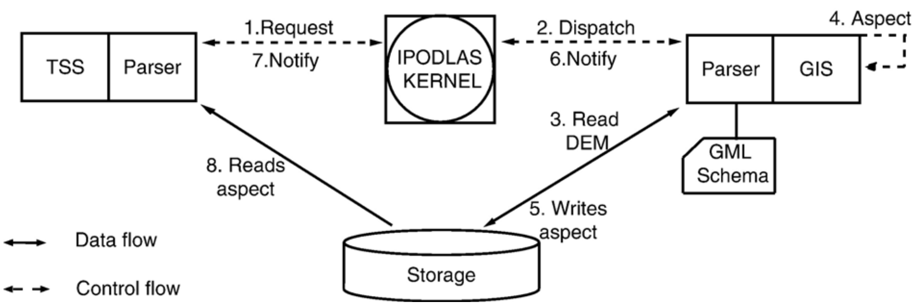

Fig. 6. Communication between the TSS and the GIS for the example of terrain aspect computation.

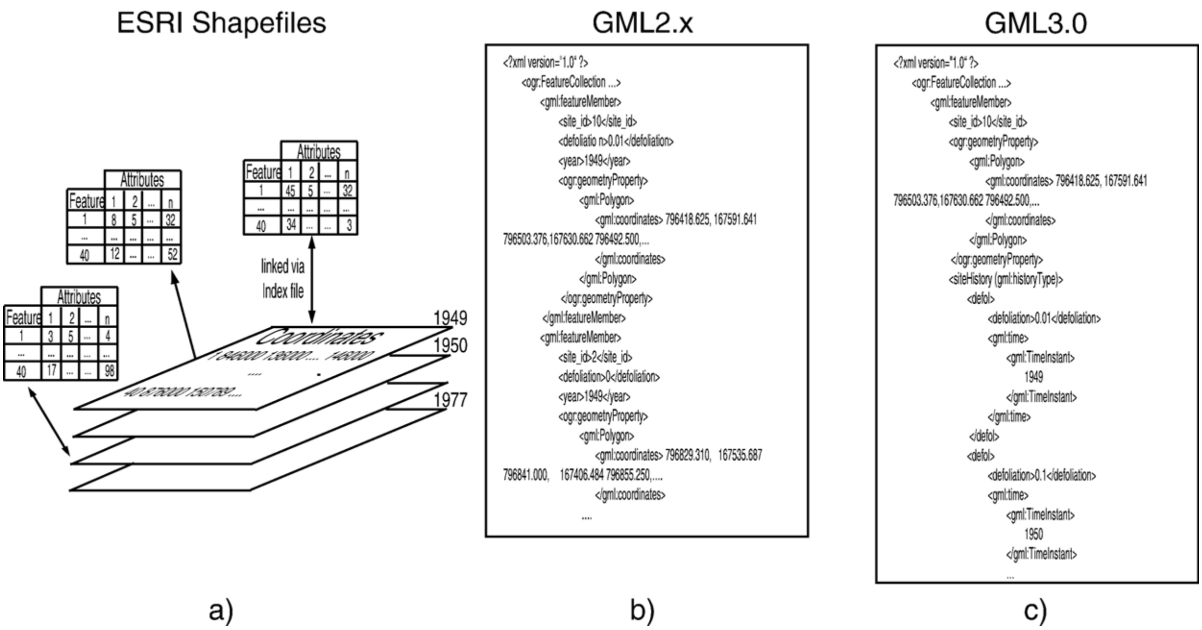

Fig. 7. a) ESRI shapefiles showing areas with LBM infestation of the years 1949 to 1977. b) The same data encoded in GML2.x. with repetitive encoding of time-invariant properties. c) In GML3.0, opposed to GML2.x, time-invariant properties are encoded only once while the time-variant properties are represented in *history* elements containing different values for each year.

the expressiveness of the data structure. The standard GIS representation of the LBM topic is that for each year a dataset exists in a snapshot-like fashion containing the information about the study area (cf. Fig. 7a). The temporal elements of GML 3 enrich the data structure to entail dynamic subsets of properties. A research area of the LBM case study is represented by a *DynamicFeature.* Time-invariant properties of a study area are for example its location, perimeter, and the coordinate system. The time slices comprise time-varying properties such as defoliation values, the amount of LBM larvae, and the year (cf. Fig. 7c). The use of the temporal features of GML3 leads to a more economic representation (Lake et al., 2004) due to the concentration of often voluminous geographic data in only one place. On the other hand, this representation supports a more object-based view of the research area, which is mapped here as one object with time-invariant properties and series of time-varying properties.

6. Discussion

6.1. The IPODLAS approach

To challenge the capabilities of IPODLAS to be able to deal with dynamic and cross-scale processes, case studies from different domains with diverse user types provide data from dynamic processes and different simulation models which act in space and time. Since simulation models acting on different scales can be coupled, the representation of cross-scale process is supported. In this project where integration of legacy systems is an important part of the development, the functionality listings are a concise and structured instrument to determine and evaluate integration efforts of existing and lacking functionality required by the users. Use of the structured approach of the UP (Jacobson et al., 1999) is a formal and transparent way for both users and developers to support the determination and description of user requirements and to move from requirements to a software system.

6.2. Software architecture

Due to the modular architecture of the system a stepwise refinement and enhancement of the system can be achieved, which allows for separate development of different aspects and therefore a smooth interaction of subsystems that are in different phases of their development. Another benefit of a modular design is the enhanced reusability (Preston et al., 2003), extensibility, and scalability of the system (Bergmann et al., 2000a; Wang, 2000). Similar to Bergmann et al. (2000b) and Bernhard and Krueger (2000) the layered architecture limits the interdependencies between the subsystems.

Additionally, as in Bergmann et al. (2000b), a central coordination process synchronizes the exchange of information between the subsystems.

6.3. Coupling TSS, GIS, and VR

Current approaches to coupling TSS and GIS and coupling GIS and VR are that GIS provide a platform for data integration, model parameter determination and cartographic visualization. TSS provides temporal capabilities and allows GIS to go beyond inventory and thematic mapping (Sui and Maggio, 1999). VR offers realistic representation and interactive exploration (Camara et al., 1998). By combining all *three* systems TSS, GIS, and VR IPODLAS moves a step further towards tool coupling following the hierarchical typology of integration of Brandmeyer and Karimi (2000), which is a networked modeling framework having integral subsystems wrapped within a common user interface. Within the framework subsystems share data and storage, and the common user interface supports seamless access to the functionality of all subsystems (Brimicombe, 2003).

6.4. GML

The encoding of the information exchange between the GIS subsystem and the central coordination process through GML is independent of the platform, operating system, language, and the data transfer protocol. Parsers can validate data structure as defined by the XML Schema. GML is an open structure providing the possibility of further enhancement. However, an increased data volume due to the tag structure has to be accepted. Compared to previous versions, GML 3.0 provides extensions covering events, histories, and timestamps (Lake, 2001; Lake et al., 2004). This offers the required data structures to prevent loss of semantics and enrich and facilitate the information exchange between the GIS and the TSS subsystem.

6.5. Lessons learned

The prototype simulating the '*intelligent tree*' showed that the file-based information exchange synchronized with semaphore files is straightforward, but becomes quickly complicated when the synchronization is complex (cf. Section 5.1.1.). The main outcome of the Cross-Implementation approach was that both subsystems (TSS and GIS) deal well with the problems they are designed for while problems occur when conducting research not explicitly supported by the systems (cf. Section 5.1.2.). This confirms the hypothesis that when doing joint research each subsystem can bring in its strengths and avoid its weaknesses. Therefore, IPODLAS can benefit from the complementary capabilities of the respective subsystems (Isenegger et al., 2004).

7. Conclusion and outlook

This paper presents the software architecture of the IPODLAS project, which aims to bring two different views and conceptualizations of views of the world – the spatial and the process-oriented – to closer proximity. Developing use cases within three diverse case studies and the derivation of listings of functionality are a systematic means to capture functional specification of requirements. To achieve the highest level of integration according to the classification of Lilburne (1996) the software architecture must be refined further to fully integrate the user interface, data and functionality of the subsystems TSS, GIS, and VR. The future of IPODLAS development is the component-based paradigm, aiming for interoperable components with exposed interfaces and hidden implementations using component-oriented middleware technologies such as CORBA (OMG, 1999), DCOM (Sessions, 1998) or EJB (SunMicroSystems, 2001). Much of the kernel functionality can be provided by an application server, which receives requests and distributes tasks to the appropriate subsystem(s), while XML-RPC (UserLand Software, 2003) or SOAP (W3C, 2003) can be used to exchange information between the subsystems. The same criteria apply to offering the functionalities of IPODLAS for geospatial services as for standard web services. Thus, IPODLAS must provide both a catalog service with metadata describing the services offered and the interfaces themselves on the syntactic and the semantic level and, furthermore, must support access via HTTP[6] and standards such as WSDL[7], UDDI[8] and SOAP to ensure interoperability (Riedemann and Timm, 2003).

Acknowledgements

This research was partially supported by the Swiss National Science Foundation under contract no. 4048-064432. The IPODLAS project is part of the National Research project NRP 48 "*Landscape and Habitats in*

[6] Hypertext Transfer Protocol.

[7] Web Service Description Language.

[8] Universal Description, Discovery and Integration.

the Alps" (NFP48, 2004) of the Swiss National Science Foundation (SNSF, 2005). The helpful comments by two anonymous reviewers are gratefully acknowledged.

References

Allgöwer, B., Fischlin, A., Frei, U., 2003. Use case approach road map, Internal report. http://bscw.geo.unizh.ch/bscw/bscw.cgi/d78284/use%20case%20approach%20roadmap.pdf (accessed October 10, 2005).

Aspinall, R., Pearson, D., 2000. Integrated geographical assessment of environmental condition in water catchments: linking landscape ecology, environmental modelling and GIS. Journal of Environmental Management 59 (4), 299–319.

Baltensweiler, W., Fischlin, A., 1988. The Larch Budmoth in the Alps. Dynamics of Forest Insect Populations. Plenum, New York.

Bennett, D., 1997. A framework for the integration of geographical information systems and modelbase management. International Journal of Geographical Information Science 11 (4), 337–357.

Bergamin, J., 2004. Schnelle Datenübertragung für verteilte Simulation und Visualisierung. MSc Thesis, Swiss Federal Institute of Technology, Zürich, Switzerland, http://bscw.geo.unizh.ch/pub/bscw.cgi/d77772/datenAT_SimuVisu_DABergamin_04.pdf (accessed October 10, 2005).

Bergmann, A., Breunig, M., Cremers, A., Shumilov, S., 2000a. Towards an interoperable open GIS. In: Norrie, M., Laurini, R., Spaccapietra, S. (Eds.), Proceedings International Workshop on Emerging Technologies for Geographical Information Systems for Geo-based Applications, Ascona, Switzerland, May 22–25, pp. 283–296.

Bergmann, A., Breunig, M., Cremers, A., Shumilov, S., 2000b. A component based, extensible software platform supporting interoperability of GIS applications. Proceedings of the Umweltinformatik 2000 — Computer Science for Environmental Protection. Metropolis-Verlag.

Bernhard, L., Krueger, T., 2000. Integration of GIS and spatio-temporal simulation models: interoperable components for different simulation strategies. Transactions in GIS 4 (3), 197–215.

Brandmeyer, J.E., Karimi, H.A., 2000. Coupling methodologies for environmental models. Environmental Modelling and Software 15 (5), 479–488.

Brimicombe, A., 2003. GIS, Environmental Modelling and Engineering. Taylor & Francis, London. 312 pp.

Buehler, K., McKee, L., 1998. The OpenGIS Guide. Open GIS Consortium Technical Committee. Wayland.

Camara, A.S., Neves, J.N., Muchaxo, J., Fernandes, J.P., Sousa, I., Nobre, E., Costa, M., Mil-Homens, J., Rodriques, A.C., 1998. Virtual environments and water quality management. Journal of Infrastructure Systems 4 (1), 28–36.

De Vasconcelos, M.J.P., Goncalves, A., Catry, F.X., Paul, J.U., Barros, F., 2002. A working prototype of a dynamic geographical information system. International Journal of Geographical Information Science 16 (1), 69–91.

Duchaineau, M., Wolinski, M., Sigeti, D., Miller, M., Aldrich, C., Mineev-Weinstein, M., 1997. ROAMing terrain: real-time optimally adapting meshes. IEEE Visualization, vol. 97. Phoenix, USA, pp. 81–88.

Fedra, K., 1993. Gis and Environmental Modeling. Environmental Modeling with GIS. Oxford University Press, New York, pp. 35–50.

Fedra, K., 1996. Distributed Models and Embedded GIS: Integration Strategies and Case Studies. GIS and Environmental Modeling: Progress and Research Issues. GIS World Books, Fort Collins, CO, pp. 413–418.

Finney, M., 2004. FARSITE, fire area simulator—model development and evaluation. Research paper RMRS; RP-4. U.S. Dept. of Agriculture, Forest Service, Rocky Mountain Research Station, Ogden, UT (324 25th St., Ogden 84401), 47 pp.

Fischlin, A., 1982. Analyse eines Wald–Insekten-Systems: Der subalpine Lärchen-Arvenwald und der graue Lärchenwickler Zeiraphera diniana GN. (Lep., Tortricidae). Diss. ETH Thesis, Swiss Federal Institute of Technology, Zürich, Switzerland. 294 pp.

Fischlin, A., 1983. Modelling of Alpine valleys, defoliated forests, and larch bud moth cycles: the rôle of migration. In: R.H. Lamberson (Ed.), Mathematical models of renewable resources. Humboldt State University, Mathematical Modelling Group, University of Victoria, Victoria, B.C., Canada, pp. 102-104.

Fischlin, A., 1991. Interactive modeling and simulation of environmental systems on workstations. In: Möller, D.P.F., Richter, O. (Eds.), Analysis of Dynamic Systems in Medicine, Biology, and Ecology. Informatik-Fachberichte. Springer, Berlin, pp. 131–145. a.o.

Fischlin, A., Baltensweiler, W., 1979. Systems analysis of the larch bud moth system: Part I. The larch–larch bud moth relationship. Mitteilungen der Schweizerischen Entomologischen Gesellschaft 52, 273–289.

Fischlin, A., Price, B., D. Isenegger, Porchet, P., Allgöwer, B., Frei, U., 2002. Use Case TreeGrowth for IPODLAS Prototype 0.0, Internal report. Workshop Nov 2002, internal report, http://bscw.geo.unizh.ch/pub/bscw.cgi/d77716/IPODLAS_UC_TreeGrowth_v0.6.pdf (accessed in October 10, 2005).

Ghezzi, C., Jazayeri, M., Mandrioli, D., 2003. Fundamentals of Software Engineering. Prentice Hall, Upper Saddle River, N.J. xx, 604 pp.

Giorgetta, F., 2002. Integration von Raum und Zeit in ein Landschaftsanalyse und Simulationssystem: Systemtheoretische Grundlagen und Evaluation gängiger Systemsimulationsprogramme. Semesterarbeit Thesis, Systems Ecology, Institute of Terrestrial Ecology, Swiss Federal Institute of Technology, Zurich, Switzerland, 54 p, http://bscw.geo.unizh.ch/bscw/bscw.cgi/d77722/fgiorgetta_2002.pdf (accessed October 10, 2005).

Goodchild, M., Steyaert, L., Parks, B., 1996. GIS and environmental modeling : progress and research issues. GIS World Books, Fort Collins, CO. xvii, 486 pp.

Hoheisel, A., 2002. SWIM meets XML— The Man Model Measurement (M3) project. http://mmm.first.fhg.de/papers/hoheisel_m3_iemss2002.pdf2002(accessed October 10, 2005).

Huang, B., Jiang, B., Li, H., 2001. An integration of GIS, virtual reality and the Internet for visualization, analysis and exploration of spatial data. International Journal of Geographical Information Science 15 (5), 439–456.

IEEE, 1990. IEEE Standard Computer Dictionary: A compilation of IEEE Standard Computer Glossaries. Institute of Electrical and Electronics Engineers, Inc., New York.

IEEE, 2000. 1516-2000 IEEE standard for modeling and simulation (M&S) High Level Architecture (HLA) — framework and rules. http://standards.ieee.org/catalog/olis/compsim.html (accessed May 1, 2005).

Isenegger, D., Price, B., Wu, Y., 2004. Crossimplementation of wildland fire and larch bud moth models in GIS and Ramses, Internal report. http://bscw.geo.unizh.ch/pub/bscw.cgi/d77674/x_implementation.pdf (accessed October 10, 2005).

ISO/TC, 2004. International Organization for Standardization/Technical committees 211. http://www.isotc211.org/ (accessed October 10, 2005).

Jacobson, I., Booch, G., Rumbaugh, J., 1999. The Unified Software Development Process. Addison-Wesley, Reading, Massachusetts.

Lake, R., 2001. GML 2.0 — Enabling the Geo-spatial Web. http://www.geojava.net/company/galdos/articles/GML3.htm (accessed October 10, 2005).

Lake, R., Burggraf, D., Trinic, M., Rae, L., 2004. GML Geography Mark-Up Language. John Wiley & Sons, Southern Gate, Chichester, West Sussex.

Leclercq, E., Benslimane, D., Yetongnon, K., 1996. A distributed object architecture for interoperable GIS. Ninth International Conference on Parallel and Distributed Computing Systems.

Lilburne, L., 1996. The integration challenge. Proceedings 8th Annual Colloquium of the Spatial Information Research Centre, Dunedin, New Zealand, pp. 85–94.

Lindstrom, P., Koller, D., Ribarsky, W., Op den Bosch, H., Hodges, L., Faust, N., 1997. An integrated Global GIS and Visual Simulation System, GVU Technical Report 97-07. Atlanta., ftp://ftp.gvu.gatech.edu/pub/gvu/tech-reports/1997/97-07.pdf (accessed October 10, 2005).

Meyer, A., Neyret, F., Poulin, P., 2001. Interactive rendering of trees with shading and shadows, Eurographics Workshop on Rendering. London, England. http://artis.imag.fr/Publications/2001/MNP01/MNP01.pdf (accessed October 10, 2005).

Neteler, M., Mitasova, H., 2002. OPEN SOURCE GIS: A GRASS GIS Approach. Kluwer Academic Publishers, Boston.

NFP48, 2004. National Research Project NFP 48, Landschaften und Lebensräume der Alpen, National Research Project NFP 48.

Nyerges, T., 1993. Understanding the Scope of GIS: Its Relationship to Environmental Modeling. Environmental Modeling with GIS. Oxford University Press, New York, pp. 75–107.

OGC, 1999. The OpenGIS Abstract Specification Overview, Version 4. http://www.opengis.org/techno/specs.htm (accessed October 10, 2005).

OGC, 2003. OpenGIS Geography Markup Language (GML) Implementation Specification, version 3.0. http://www.opengeospatial.org/specs/?page=specs (accessed October 10, 2005).

OGC, 2005. OGC — Open Geospatial Consortium. http://www.opengeospatial.org/2005(accessed October 10, 2005).

OMG, 1999. CORBA components and component model, document orbos/99-02-05. http://www.omg.org/ (accessed October 10, 2005).

Pajarola, R., Widmayer, P., 2001. Virtual geoexploration: concepts and design choices. International Journal of Computational Geometry and Applications 11 (1), 1–14.

Pang, M.Y.C., Shi, W.Z., 2002. Development of a process-based model for dynamic interaction in spatio-temporal GIS. Geoinformatica 6 (4), 323–344.

Peuquet, D.J., Niu, D.A., 1995. An Event-Based Spatiotemporal Data Model (Estdm) for Temporal Analysis of Geographical Data. International Journal of Geographical Information Systems 9 (1), 7–24.

Preston, M., Clayton, P., Wells, G., 2003. Dynamic run-time application development using CORBA objects and XML in the field of distributed GIS. International Journal of Geographical Information Science 17 (4), 321–341.

Price, B., Isenegger, D., Wu, Y., 2005. Use Case Larch Budmoth Expert 2, Internal report. http://bscw.geo.unizh.ch/pub/bscw.cgi/d77703/use_case_LE2_v1_2.pdf (accessed October 10, 2005).

Raper, J., Livingstone, D., 1995. Development of a geomorphological spatial model using object-oriented design. International Journal of Geographical Information Science 9 (4), 359–383.

Riedemann, C., Timm, C., 2003. Services for data integration. Data Science Journal 2, 75–83 (Spatial data usability special section).

Rothermel, R., 1972. A mathematical model for predicting fire spread in wildland fuels. Internal report., USDA Forest Service, Intermountain Forest and Range Experiment Station.

Savary, L., Zeitouni, K., 2003. Spatial data warehouse — a prototype. Electronic Government, Proceedings, vol. 2739, pp. 335–340.

Schöning, R., 2000. Modellierung des potentiellen Waldbrandverhaltens mit einem GIS. Msc Thesis Thesis, University of Zürich, Zürich, Switzerland, http://www.geo.unizh.ch/gis/research/geoprocessing/gp-abst29.shtml (accessed October 10, 2005).

Schulze, T., Wytzisk, A., Simonis, I., Raape, U., 2002. Distributed spatio-temporal modeling and simulation. In: Yuecesan, E., Chen, C., Snowdon, J., Charnes, J. (Eds.), Winter Simulation Conference, San Diego, CA, pp. 695–703.

Sessions, R., 1998. COM and DCOM. John Wiley Press, New York.

SNSF, 2005. Swiss National Science Foundation, http://www.snf.ch/default_en.asp (accessed October 10, 2005).

Stevens, W., 1990. UNIX Network Programming. Prentice Hall P T R, Englewood Cliffs, NJ.

Sui, A., Maggio, R., 1999. Integrating GIS with hydrological modeling: practices, problems and prospects. Computers, Environment and Urban Systems 23, 33–51.

SunMicroSystems, 2001. Enterprise JavaBeans technology. http://java.sun.com/products/ejb/ (accessed October 10, 2005).

UserLand Software, 2003. XML-RPC Home Page, http://www.xmlrpc.com/ (accessed October 10, 2005).

Vckovski, A., 1998. Interoperable and distributed processing in GIS. Taylor & Francis, London. xvii, 230 pp.

W3C, 2003. SOAP Version 1.2. http://www.w3.org/TR/soap/ (accessed October 10, 2005).

Wang, F., 2000. A distributed geographic information system on the common object request broker architecture. GeoInformatica 4 (1), 89–115.

XML, 2004. Applying XML and Web Services Standards in Industry. http://www.xml.org (accessed October 10, 2005).

Yates, P., Bishop, I., 1997. A Method for the Integration of Existing GIS and Modeling Systems, GeoComputation. University of Otago, New Zealand, pp. 191–197.

Zaslavsky, I., Marciano, R., Gupta, A., Baru, C., 2000. XML-based Spatial Data Mediation Infrastructure for Global Interoperability, 4th Global Spatial Data Infrastructure Conference, Cape Town, South Africa, March 13–15. http://www.npaci.edu/DICE/Pubs/gsdi4-mar00/gsdi_iz.html (accessed October 10, 2005).

Zeigler, B.P., 1976. Theory of Modeling and Simulation. Wiley, New York. xxii, 435 pp.

Zeigler, B.P., 1990. Object-Oriented Simulation with Hierarchical, Modular Models: Intelligent Agents and Endomorphic Systems. Academic Press, Boston. xvii, 395 pp.