DIPLOMA THESIS

Interactive online visualisation of cartographical data
An example of air pollution simulation in Europe

A diploma thesis submitted in partial fulfilment of the requirements for the degree of Dipl.-Ing. (FH) for Computersimulation at the University of Applied Sciences St. Pölten

Under the scientific supervision of

Wolfgang Schöpp (IIASA)
and
Matthias Wolf (FH St. Pölten)

by

Rafal Cabala
si011004

St. Pölten, 5th Mai 2005

To the entire staff of the Transboundary Air Pollution Program (TAP) of the International Institute of Applied Science Analysis.

I would like to express my gratitude for the opportunity to develop the concept for my diploma thesis, and to complete the work during a most worthwhile internship from August through December 2004.

Generous support and valuable criticism made it possible for me to work on my own thesis as part of a trustily qualified team.

For this opportunity I wish to express my sincere.

THANKS!

Abstract

This thesis presents a software program developed to visualize air pollution indicators for mitigation scenarios developed by International Institute for Applied Science Analysis (IIASA) integrated assessment model, Regional Air Pollution Information and Simulation Web (RAINS Web). The software is later called RAINS IMPACT. It uses data on deposition and concentrations of pollutants in Europe, combines it with ecosystems and human sensitivities to air pollution and creates policy-relevant indicators. Next, these indicators can be presented in graphical form as a series of maps. The system contains several possibilities to define and extract the maps, and export them in a standardized output format. The software is fully operational and will be soon incorporated into the RAINS Web model. RAINS IMPACT uses the state-of-the art Web programming technology. Because of its flexible architecture, it can be easily modified for new tools and standards that are likely to emerge within the years to come.

Content

1 Background and the scope of the study

One picture is worth a 1000 words…
Proverb

A person who needs to read a document of several hundred pages and examine huge tables with diverse information about specific geographical regions for professional reasons has a very hard job. Thus computer tools are being developed to make such an analysis easier. One of the most important tools for than type of work is data visualization. With these tools it is possible to present sets of numbers in a form of easy to read graphs or maps that enable to draw the right conclusions.

The Internet holds a great potential as a distribution medium for maps and geographic information. Its abilities to connect directly to source data on servers and to reach a global audience at the same cost as reaching a single person have made Web Geographical Information Systems (GIS) a rapidly growing industry. However, currently available Web mapping software is very expensive and leaves much to be desired: map quality is in many cases poor, interactivity capabilities are limited, and only simple database queries are allowed.

Visualization tools are of particular importance for many scientific models, where what matters is the spatial distribution of input data and results. This is also true for the IIASA[1] RAINS model[2], which is used for integrated assessment of air pollution control strategies at a continental level (Europe, Asia), for individual countries (the Netherlands, Italy), or even for sub-national regions. Recently a Web version of the model was prepared. The model enables simulations via the Internet, of various scenarios for reducing air emissions in a selected region. A short description of the model is provided in the appendix. However, until recently, presentation of model output in terms of the spatial distribution of environmental impacts requires intensive data processing via specialized program available only to the Transboundary Air Pollution (TAP) staff[3]. Thus, implementation of visualization capabilities into RAINS Web were regarded as one an important priority task in further model development.

1 "IIASA" is the International Institute for Applied Systems Analysis. The Institute located in Laxenburg Lower Austria (A-2361 Laxenburg, Austria), is supported by non governmental organizations from 17 countries. Its main task is conducting of interdisciplinary scientific studies on environmental, economic, technological and social issues in the context of the human dimensions of global change.

2 Compare Amann et al., 2002 – see appendix.

3 Currently maps can be prepared with Perl script, which is not fully integrated into RAINS WEB. Graphics capability also exists in the PC version of RAINS (the so-called DEP module – compare Amann et al., 2000 – see appendix)

This diploma thesis is the response to that priority task. The topic was formulated during the author's internship at IIASA between August and December 2004. Terms of reference of the task, a general methodology, together with theoretical and practical issues that needed to be addressed in the study are discussed in Section 3. Section 4 presents an overview of Internet development formats and technologies currently available for the preparation of the required software. Advantages and disadvantages of possible approaches and configurations are discussed and a justification for the selected set of basic tools that was chosen is provided. Section 5 gives detailed information about open source programming, which is one of the requirements for visualization software. Section 6 investigates of online mapping technologies. Section 7 describes the reasons for the software selected and Section 8 presents the details of the adopted solution, and demonstrates the example applications. Finally, Section 9 summarizes the main features of the software and specifies the needs for further modifications and extensions.

2 Problem formulation and major tasks

The aim of this thesis is to add interactive graphics capability to the RAINS Web model. The model should be able to calculate environmental impact indicators for a selected emission scenario simulated by the user and to visualize them through preparation of appropriate maps in an interactive way. In the next step the software should enable the storing produced output and export it for use in other applications in the formats appropriate for handling graphical information. Solving such problems required addressing several theoretical and practical issues. They are discussed below.

The most important **theoretical tasks** were the following:

- Investigating the possibilities of using Macromedia Flash ® or Scalable Vector Graphic® (SVG) or JAVA Applets as tools for extending the RAINS model with interactive graphics;
- Analysing advantages and disadvantages of these Internet development technologies according to cartographical database information;
- Defining the best technology for visualization of cartographical database information as maps;
- Developing an appropriate client-server protocol for exchange of information;
- In software design applying methodologies that can be adapted to the expected near-term changes in simulation environments and standards expected for Web applications (structured modelling, Extensible Markup Language (XML), etc.);

In fact, addressing the last issue can be regarded as the main research challenge of the thesis. It can be formulated as the following question:

> *"Are the solutions used for visualization of cartographical database information as maps or business graphs future–oriented?"*.

Practical (technical) tasks that needed to be resolved were the following:

- Assuring that the software solution is compatible with the IIASA software licensing policy (for broader discussion see chapter 5);

- Assuring the compatibility of the new software with the existing online model (RAINS Web)[4];
- Developing the possibility for production of printable maps with one of the following extensions: PostScript (PS), Encapsulated PostScript (EPS) or Portable Document File (PDF);
- Developing possibility for downloading the graphics produced;
- Providing tools for changing the layout, color definition, and limits of maps' legends depending on the needs of the client;
- Checking the feasibility of serial production of the output, for instance, preparation of maps with several impact indicators for one year in one step, preparation of maps for one indicator for several years, or both;
- Preparing examples of European maps demonstrating environmental impacts of emission scenarios. Geographic information should be imported from a GIS system (Arc View (Info)® or MapInfo®);
- Exploring the behaviour of the routine with the most popular browsers (Internet Explorer, Netscape 7.1+, Opera 7.x+ and Mozilla - Firefox);

According to the best practices described by Mag. Edith Huber[5] the **research strategy** adopted in our study was the **quality content analysis**. This means that we tried to combine the best theoretical information currently available with existing practical experience of the author and his supervisors. Within the thesis we have performed an extensive review of relevant literature (books, journals, IIASA reports, as well as the Internet Websites (*http://www.oracle.com, http://www.xml.org, http://www.adobe.com, http://www.sun.com*, etc.). We combined the theory with practical knowledge and "hands on" experience gained during the internship at IIASA.

4 This means that it should be based on JAVA servlets that run on IPlanet (for yet another appliance on Tomcat 5.x). The database system is ORACLE, but it is also compatible to PostGreSql.
5 "Diplomarbeitsleitfaden", St. Pölten 2004

3 General information on online mapping

3.1 Basic ideas

Making maps available on the Internet can be as simple as designing a map with your favorite desktop GIS and uploading the image to a Web server. But the most interesting Web maps are generated "on the fly" in response to a user-initiated request.

Internet map servers are software components that orchestrate the dynamic creation of maps and associated cartographic elements (scale bar, legend, reference map, etc.) from spatial data sources such as a spatial database, Geography Markup Language (GML) documents, proprietary data formats or real time data from roving Global Positioning System (GPS) receivers. Many technical strategies for Web mapping can be deployed, but the basic scenario involves extracting the coordinates of spatial features from a data source and converting them to a raster (Portable Network Graphic - PNG) or vector (SVG) image that is returned to a Web browser. This process can be implemented with a wide range of open-source programming tools, including Perl, Hypertext Pre Processor Language (PHP), JAVA (not open-source, but without costs), Python, etc. Most map servers include client-side interfaces for interacting with server generated maps. Highly interactive client-side interfaces can be implemented with Macromedia Flash, JAVA Applets, JavaScript or SVG.

3.2 Concise technical information

Technically, there are two ways to provide Web-based interactive cartography:

- As bitmap images produced on-the-fly by a server specializing in geographic processing;
- As images drawn by the browser through specific interpreters;

In the first case the browser displays Graphic Interchange Format (GIF) or Joint Photographic Experts Group (JPEG) images. This technique is the most reliable. It guarantees that the Web-based cartography application will be viewable immediately by all browsers, except those that cannot read GIF or JPEG files (if such browsers even exist today). It offers the least viewing and printing quality, slow display speed for large sizes and interactivity. Strangely, it is nevertheless the most expensive! It is offered by all major GIS editors, through the installation of a specific application on the Internet server. Of

course, the user either has a strong host server on which he can install his own software, or he relies on a hosting provider specializing in cartography servers.

In the second case, several solutions are possible. Browser specific interpreters (DirectX for the Personal Computer (PC) version of Internet Explorer) can be employed, the JAVA virtual machine, when installed, or additional client program (plug-ins) of varying sizes and levels of cartographic specialization can be developed. Finally, standard integrated SVG interpretation is possible. If the browser is able to send vector image data to a printer port (including the Acrobat Distiller), the images are called vector images and these are high-quality exports or printouts. If this is not possible, bitmap images are drawn on demand by the browser. In fact, an image is always converted to bitmap when it is displayed or printed, but it should loose its vector data source as late as possible. Therefore this technique is more widely used. It is also relatively new, and the subject of many discussions. First, it pits the specialized technology of major GIS editors (Esri, Mapinfo, Autodesk, etc.) against general purpose vectorial formats (Flash, SVG). Secondly, it sparks lively debates between "Flash-ers" and "SVG-ists".

Since the scientific technique relies on a client browser, an entity that is by definition less uniform and controllable than a server, it must abide by three simple rules if it is to reach a wide audience. The rules are as follows:

- Guaranteed compatibility of the viewing technique (plug-in, JAVA) with all browsers, or knowledge of the present state of compatibility and future involvement of concerned editors. Well-placed contestants include major GIS editors, who have been developing and distributing such plug-ins for over 3 years, and Macromedia, which reached this priority objective on all counts with the release of version 6 for Linux at the end of 2002. As for SVG, Corel was the most dynamic player in late 2002 and early 2003, with the release of a player and a development tool. The SVG developer community is waiting impatiently for Adobe's reaction. The latest version 6 of its player was published in early 2004.
- A reasonable small plug-in size (under 1Mb, if possible). Weighing in at 400Kb, Flash leads the way; other plug-ins are rarely smaller than 2Mb. JAVA solutions use the virtual machine of the same name, which is not always included with the browser and represents over 10Mb. In addition, JAVA applications are not always stable, to say the least, and the entire application must be reloaded each time it is used. At the same time, plug-ins once installed support many basic functions once and for all. As for the integration of vectorial features in browsers, SVG is the standard recommended by W3C but Microsoft continues to favour Vector Markup Language (VML) and has no plans to make a change. Mozilla planned to include

standard SVG playing. But this goal has not yet been reached and Mozilla's market share remains limited. If Microsoft finally includes SVG, it will not be done earlier than one year. This mean that it will take at least 2-3 years before the new browser version is sufficiently available.

- A widely distributed plug-in. Figure 1 presents the distribution of plug-ins used in the USA. The Figure 1 presents the results of a quarterly survey organized by an independent institute (at the beginning of 2004). Methodology: quarterly survey of 2,000 people who tested on screen whether they saw the same content in each of the above formats.

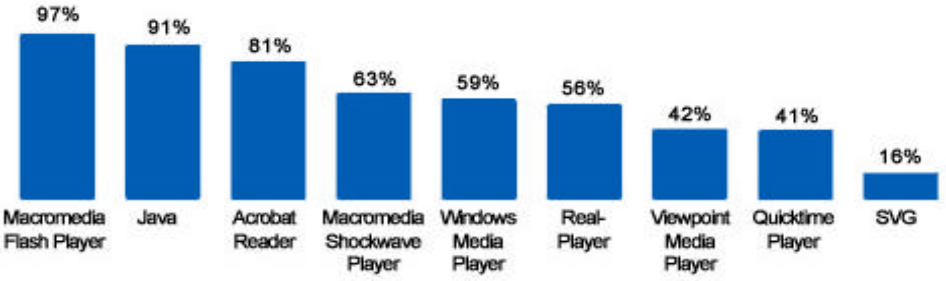

Figure 1: Plug-in statistic (USA); Copyright 2004 The NPD Group, Inc.

http://www.npd.com

3.3 Working with open-source software

Deploying open-source software offers and poses many challenges. Typical open-source installation involves downloading source code for the target operating system, identifying and downloading other required software components, configuring the desired features and compiling the applications. This process is easy and routine for many developers, but complex to users with little programming experience. Most mature open-source software is well supported and includes thorough installation instructions. Any motivated power user of proprietary, closed source, Graphical User Interface (GUI)-driven GIS software who can follow a technical "cookbook" will be able to install and use open-source software.

A significant challenge faced when implementing an advanced open-source Web GIS is the breadth of technical skills required and the logistics of the interaction of many complex applications. Designing Web-based geospatial solutions requires a thorough understanding of core Web technologies, including the configuration and secure management of Web servers. Spatial information management expertise is another skill needed to create the geoprocessing steps required to solve spatial problems in a distributed environment with different data formats over varying transport media.

4 Available formats and technologies for the task

For better understanding of the online visualization solutions considered in our study, the basics of formats and technologies used will be discussed, together with their areas of application, advantages and disadvantages.

4.1 General programming standards

The following Web programming standards are the most important components of the development of Web applications. Nowadays no modern Web project can be developed without them.

DOM

Document Object Model (DOM) is a form of representation of structured documents as an object-oriented model. DOM is the official World Wide Web Consortium (W3C) standard for representing structured documents in a platform- and language-neutral manner.

DOM was initially supported by Web browsers to manipulate elements in a HyperText Markup Language (HTML) document. DOM was a way of dynamically accessing and updating the content, structure and style of documents. Owing to incompatibilities in the DOM implementation between different browsers, the W3C came up with standard specifications for DOM.

DOM does not put restrictions on the document's underlying data structure. A well-structured document can take the tree form using DOM. Most XML parsers have been developed to make use of the tree structure. Such an implementation requires that the entire content of a document be parsed and stored in memory. Hence, DOM is best used for applications where the document elements have to be accessed randomly and manipulated. For XML-based applications which involve a one-time selective read/write per parse, DOM requires a considerable memory overhead.

XML

Extensible Markup Language (XML) is a W3C recommendation for creating special-purpose markup languages. Markup language is a form of text encoding that represents the text itself as well as details of its structure and appearance of the text. A modern one

with widespread use is HTML. Markup languages are used by the publishing industry to share printed works among authors, editors, and printers. XML is capable of decrypting a wide variety of data. Its primary purpose is to facilitate the sharing of structured text and information across the Internet.

Languages based on XML, i.e., Resource Description Framework (RDF), RDF Site Summary (RSS) or SVG, are themselves decrypted in a formal way, allowing programs to modify and validate documents in these languages without prior knowledge of their forms.

Advantages

- Simultaneous human- and machine- readable format;
- XML is self-documenting in that it describes the structure and field names as well as specifying values;
- The ability to represent the most general computer science data structures;
- Support for Unicode, an international standard whose target is to provide the means to encode the text of every document that is stored in computers;
- Strict syntax makes the parsing algorithm simple, fast, and efficient;

XML also offers several benefits for widely used formats for document storage and processing, both online and offline:

- Hierarchical structure suitable for most types of documents;
- Robust and logical format based on international standard;
- Plain text files, independent from licenses or restrictions;
- Platform-independent;
- Extensive experience and software available, since it has already been in use for long time;

Disadvantages

- XML syntax is fairly verbose and partially redundant. It can make XML difficult to apply in cases where bandwidth is limited (compression can reduce this problem);
- XML still often requires further parsing to extract individual values;
- No facilities for randomly accessing or updating only portions of a document;
- Modelling overlapping (non-hierarchical) data structures requires extra effort;

JavaScript and ECMA Script

JavaScript is an object-based scripting programming language. Some experts are of the opinion that JavaScript has syntax close to that of Sun Microsystems' JAVA programming language. However, except for the name and syntax, the language has more in common with Self programming language than with JAVA.

Names convention

The first name of this programming language was LiveScript, which changed to JavaScript as Netscape was including support for JAVA technology in its Netscape Navigator browser. There is no real relationship between JAVA and JavaScript, their similarities are mostly in syntax, the semantics are quite different, and their object models are unrelated and in the most part incompatible.

Usage

JavaScript embedded in a Web browser connects through interfaces called DOM to applications, especially to another server side and the client-sides Web browser of Internet applications. The powerful dynamic Web applications are the most important reasons for usage. It may use Unicode and can evaluate regular expressions.
One major use of the Web-based JavaScript is to write functions that are embedded in HTML pages and interact with the DOM of the browser to perform tasks not possible in static HTML alone, such as opening a new window, checking input values, changing images as the mouse cursor is moved over test picture, etc. Unfortunately, the DOMs of all browsers are not standardized. Different browsers expose different objects or methods to the script. It is, therefore, often necessary to write different variants of a JavaScript function for the various browsers.

Outside of the Web, JavaScript interpreters are embedded in a number of tools, i.e., Adobe Acrobat and Adobe Reader support JavaScript in PDF files.

Incompatibilities

First of all most incompatibilities are not JavaScript issues but DOM-specific. Some incompatibility issues existing across JavaScript implementation include handling of certain primitive values such as undefined, and the availability of popular methods, such as the *.pop(), .push(), .shift()* or *.unshift()* methods of arrays.
JavaScript, like HTML, is often not incompliance to W3C standards, being built instead to work with specific Web browsers. The current ECMAScript standard should be the

basis for all JavaScript implementations in theory, but in practice the Mozilla family of browsers (Mozilla, Firefox and Netscape Navigator) uses JavaScript, Microsoft Internet Explorer uses JScript, and other browsers such as Opera and Safari use other ECMAScript implementations, often with additional non-standard properties to enable compatibility with JavaScript and JScript.

Offspring

The programming language used in Macromedia Flash (called ActionScript) bears a strong resemblance to JavaScript due to their shared relationship with ECMAScript. ActionScript has nearly the same syntax as JavaScript, but the object model is dramatically different.
A novel example of the use of JavaScript are Bookmarklets, small sections of code within Web browser Bookmarks or Favorites.

JAVA Virtual Machine

JAVA Virtual Machine (JVM) is a virtual machine that runs JAVA byte code. This code is generated mostly by JAVA compilers, although the JVM has also been targeted by compilers of other languages.

The JVM is one of the most important components of the JAVA platform. The availability of JVM's on almost all types of hardware and software platforms enables JAVA to function both as middleware and a platform in its own right. Sun Microsystems created a slogan to illustrate the cross platform benefits of the JAVA language: "Write once, run anywhere".

Programs that are made to run on a JVM must be compiled into a standardized portable binary format, which comes, typically in the form of .class files. A program may consist of many classes, in which case every class will be in a different file. For easier distribution of large programs, multiple class files may be packaged together in a .jar file.

The JVM verifies the bytecode of the program before it is executed. This means that only a limited amount of bytecode sequences from valid programs can target an instruction within the same function. Because of this, the fact that JVM is stack architecture does not imply a speed penalty for emulation on register-based architectures when using a JIT compiler. In fact, code verification makes the JVM different from a classic stack

architecture whose efficient emulation with a Just-in-Time-compiler is more complicated and typically carried out by a slower interpreter.

Secure execution of remote code

Virtual machine architecture allows very fine-grained control over the actions that code within the machine is permitted to take. This allows safe execution of untrusted code from remote sources, a model used most famously by JAVA Applets (see chapter 2.1.1. - JAVA Applets). Applets run within a JVM incorporated into a user's browser, executing code downloaded from a remote HTTP server. The remote code runs in a highly restricted "sandbox", which protects the user from misbehaving or employing malicious code. Publishers can apply for a certificate with which to digitally sign Applets as "safe", giving them permission to break out of the sandbox and access the local filesystem and network, presumably under user control.

4.2 Client-side technology

SVG

Scalable Vector Graphic (SVG) is an XML markup language for describing two-dimensional vector graphic, both static and animated. It is an open standard created by the W3C, which is also responsible for HTML and Extensible HMTL (XHTML) standards.

Overview

SVG allows three types of graphic objects:

1. Vector graphic shapes (i.e., paths consisting of lines and curves and areas bounded by them);
2. Raster graphics images / digital images;
3. Text;

Graphic objects can be grouped, styled, transformed, or composed into previously rendered objects. Text can be in any XML namespace suitable to the application, which enhances search ability and accessibility of the SVG graphics.

SVG drawings can be dynamic or interactive. A rich set of event handlers such as *onmouseover* and *onclick* can be assigned to any SVG graphical object. Because of its

compatibility to other Web standards, features like scripting can be done on SVG elements and other XML elements from different namespaces simultaneously within the same Web page.

If transfer is an issue, SVG images are sometimes saved with gzip compression, in which case they may be called "SVGZ files". Because XML contains a lot of redundant data, XML tends to compress very well and these files can be much smaller.

Support for SVG in browsers and other applications

The use of SVG on the Web is in its infancy. There is a great deal of inertia from the long-time use of completely raster formats, but browser support is also patchy, with users of most browsers having to install a plug-in. Web sites that serve SVG images typically also provide the images in a raster format, either automatically by Hypertext Transfer Protocol HTTP content negotiation or allowing the user to directly choose the file.

Plug-in support

In most browsers, such as the Internet Explorer, a plug-in is needed to see SVG images in the browser window. Currently available SVG plug-ins include Adobe SVG Viewer3[6] and 6 Beta.

Native support

There are several advantages to native support. For example, there is no need for a plug-in, SVG can be mixed with other formats in a single document, and scripting between different document formats is a lot more reliable.

The Mozilla SVG Project[7] is working a native SVG support to Mozilla. A preview of the SVG support is expected to be included in the next official releases of Mozilla Suite and Mozilla Firefox.

The K Desktop Environment KDE project's Konqueror Web browser also has a fairly complete SVG implementation called ksvg[8].

SVG is natively supported in the Amaya Web browser.

[6] *http://www.adobe.com/svg/viewer/install/main.html*
[7] *http://www.mozilla.org/projects/svg*
[8] *http://svg.kde.org*

The Opera Web browser offers SVG support based on the SVG 1.1 Tiny standard.

Tools

Most of the major drawing software packages such as Adobe Illustrator and Corel Draw support SVG export. OpenOffice.org Draw 1.1 and up can also export SVG files while *SVGmaker*[9] creates SVG from standard Windows programs, including the ubiquitous Office suite. Sodipodi[10] and Inkscape[11] are two other (open-source, multi-platform) tools that use the SVG format. Sketsa[12] is another native SVG Graphics Editor.

As in XML, SVG has different kinds of standard parsers for reading and writing. An example of this kind of technology is Batik[13] by Apache. [1] Batik is a JAVA technology based toolkit for applications or Applets that want to use images in the SVG format for various purposes, such as viewing, generation or manipulation. Batik supplies a set of standard modules such as SVG Parser, SVG Generator, and SVG DOM.

Macromedia Flash

Macromedia Flash or Flash is a graphics animation program, developed by Macromedia that uses vector graphics. Vector graphics use geometrical primitives such as points, lines, curves, and polygons to represent images in computer graphics – it is used as a contrast to the term raster graphics technology, which uses pixels to represent an image. The resulting files (called Shock Wave File SWF files) may appear in a Web page to be viewed in a Web browser. A stand alone Flash player is also allowed to present them. The most common usage of Flash is in animated advertisements on Web pages and rich media Web sites.

In the latest version of Macromedia Flash (MX), this tool has been expanded beyond the simple animations of Web banner advertisement into a complete application development tool.

9 http://www.svgmaker.com
10 http://www.sodipodi.com
11 http://www.inkscape.org
12 http://www.kiyut.com/products/sketsa/index.html
13 http://xml.apache.org/batik

Language

Flash MX (the latest version) comes with ActionScript 2.0, which is in compliance to ECMAScript 4. This means that it looks more like JAVA. It can now be considered a full-fledged *object oriented programming*[14] language, including all its free-form coding styles. But this paradigm has no polymorphism, operator-overloading, any sort of classes, truly private scope, runtime-type checking.

Advantages

- Flash can be used to specify the exact position of the various page elements.
- Flash supports streaming by default.
- Flash uses vector graphic, which means no less of image quality after resizing.
- Flash players can run consistently on Windows from MS, Max OS, Linux, UNIX, Solaris, HP-UX, Pocket PC and OS/2.
- Flash allows the embedding of images, videos, sounds, and even simple HTML files.
- Embedded ActionScript language allows the creation of sophisticated applications.
- According to Macromedia, 95% of Web users have installed Flash.
- Personal Digital Assistant (PDA) and cellular phones can integrate Flash players, and implementation exists for the JAVA platform.
- The Flash plug-in is extremely fast in initializing (compared to other browser plug-ins such as QuickTime Player or JAVA).

Disadvantages

- Flash does not use browser settings for font, size, etc., so text may appear very small for some users.
- Flash content remains inaccessible to most search engines.
- Not all operating systems have a plug-in.
- Because a user agent[15] plug-in plays Flash movies, such movies have limited memory available to them. The significance of this disadvantage is reduced by the Flash player's internal memory management.
- Since Flash files do not depend on a truly open standard such as SVG, this reduces the incentive for non-commercial software to support the format.
- Older versions of Flash do not support internationalization completely.

14 Computer programming language where software is modelled as a set of objects that interact with each other.
15 is a client application used with a particular network protocol.

- Flash demands significant Common Process Unit (CPU) power to display, as it uses a very high degree of graphic abstraction that many video cards can not support.
- Flash plug-in is able to store and retrieve information on a user's computer, acting like an HTTP cookie.
- The SWF files generated by Flash present security issues. Several available commercial programs allow someone to extract graphics and sounds from a SWF file and also view the ActionScript. But the complexity of the SWF files makes the extraction mostly impossible.

Influence

The nature and popularity of Flash has had a large influence in graphic design. Its rotoscoping[16] feature led to the widespread popularity of rotoscoped vector graphics in the default pastel colors of the Flash authoring tools. Many flyers, advertisements, magazines, and even Websites that did not use Flash adopted this graphic style.

JAVA Applet

A JAVA Applet is an *Applet*[17] written in JAVA programming language. JAVA Applets run in a Web browser using JVM or a stand alone tool to test Applets (Sun's Applet Viewer).

Usage of Applets is providing features to Web applications that can not be provided by HTML. They are executed in a *sandbox*[18]. The sandbox security model provides a tightly controlled set of resources in which foreign programs can run. This program requires only a small space on the disc and a section of memory to carry out instructions. User interaction may also be allowed by sandbox, and it is also possible to prompt the user to allow or disallow certain actions as the program runs.

In the JAVA system, most Applets run in a sandbox that provides at minimum a rectangle of screen space and, optionally, some disk space and memory – of course with the user's permission. The code of the Applet is downloaded from a Web server and the browser either embeds the Applet into a Web page or opens a new window – as user

16 A rotoscope is a device that enables animators to trace live action movement, frame by frame, for use in animation. It might be called a clumsy forerunner to digital motion capture
17 An Applet is a small program that runs in the context of a larger program on a client on computer. Nowadays this refers mostly to JAVA Applets. These run in a browser.
18 A safe place, in a computer security, for running semi-trusted programs or scripts.

interface. The Applet is embedded in the Web page by making use of the HTML element `Applet`. This specifies the source, the location, and the size of the Applet. The Applet does not allow itself to be controlled with Cascading Style Sheet (CSS).

Since JAVA's bytecode is platform independent, JAVA Applets can be executed by browsers for many platforms (Windows, Linux, UNIX, Max OS or Solaris).

For Applets to be compliant with Microsoft's operating system Windows, Applet code has to be developed using JAVA compatible with Microsoft JVM. It is also possible to run another more, modern JVM on Windows that has been downloaded and installed from Sun.

4.3 Server-side technology

SQL

Structured Query Language (SQL) is the most popular computer language used to create, modify, and query databases.

A database language standard specifies the semantics of various components of a database management system (DBMS). In particular, it defines the structures and operations of a data model implemented by the DBMS, as well as other components that support data definition, data access, security, programming language interface, and data administration. The SQL standard specifies data definition, data manipulation, and other associated facilities of a DBMS that supports the relational data model.

A database language standard is appropriate for all database applications where data will be shared with other applications, where the life of the application is longer than the life of current equipment, or where the application is to be understood and maintained by programmers other than the original ones.

SQL keywords

Here some most important SQL keywords, split in to three groups are described:

Standard Data Manipulation Language (DML) elements

DML is the subset of the language used to query a database and add, update, and delete data.

SELECT – used to retrieve zero or more rows from one or more tables in a database.
INSERT – used to add zero or more rows to an explicit table.
UPDATE – used to modify the values of existing table rows.
DELETE – used to remove zero or more rows from a table.

Addendum to DML
START TRANSACTION – used to mark the start of a database transaction[19].
COMMIT – used to confirm all data changes to make them permanent.
ROLLBACK – used to discard all data changes since the last COMMIT or ROLL-BACK

Data Definition Language (DDL) elements

DDL allows the user to define new tables and associated elements.

CREATE – used to create a new object (i.e., a table or view) within the database.
DROP – used to delete an already existing object (i.e. a table or view) in the database.

Addendum to DDL
Some database systems also have an ALTER command, which is used to modify an existing object in various ways (i.e., adding columns to an existing table).

Data Control Language (DCL) elements

DCL handles the authorization aspects of data and permits the user to control who has access to see or manipulate data within the database.

GRANT – used to authorize a user to perform some operations.
REVOKE – used to remove the authorization for a user to perform some operations.

19 A database transaction is a unit of interaction with a database management system or similar system that is treated in a coherent and reliable way independent of other transactions

ORACLE database

An ORACLE database is a collection of data managed by an ORACLE database management system (DBMS).

The ORACLE DBMS is produced and marketed by ORACLE Corporation[20]. The ORACLE DBMS is extensively used by many database applications on most popular computing platforms.

Database structure

ORACLE stores data logically in the form of table spaces and physically in the form of data files. Table spaces can contain various types of segments, i.e., data segments, index segments, etc. In turn, segments are made up of one or more extension. Extensions are grouped based on contiguous data blocks. Data blocks are the basic units of data storage. On the physical level, data files are made up of one or more data blocks, where the block size can be variable.

ORACLE keeps track of data storage with the help of information stored in the system table space. This system contains the data dictionary, indexes, and clusters. A data dictionary is a special collection of tables that contains information about all user objects in the database.

Stored procedures and functions can be executed within the database. These can be developed in ORACLE's proprietary procedural extension to SQL, Procedural Language SQL (PL/SQL), or in the object-oriented language JAVA.

An ORACLE database installation traditionally comes with a default schema called "scott". After the sample tables have been created, the user can log into the database with the user "scott" and password "tiger".

Working platforms

In the past (before version 9i) ORACLE had exported its database engine to a wide variety of platforms. Recently, ORACLE has specified its database for a smaller range of operating system platforms.

20 http://www.oracle.com

PostGreSql database

PostGreSql is a free object-relational database server. It offers an alternative to other open-source database systems (such as My Structured Query Language (MySQL) and Firebird), as well as to proprietary systems such as ORACLE, Sybase, IBM's DB2 and Microsoft SQL Server.
PostGreSql database has very similar features to the ORACLE. One of the few differences is most notably the licensing policy. ORACLE is the commercial and PostGreSql is a free solution for a database environment.

JAVA Servlet

The **JAVA Servlet Application Programming Interface (API)** allows a software developer to add *dynamic* content to a Web server using the JAVA platform. The generated content is commonly HTML, but may be other file format such as XML. Servlets is a dynamic Web content technology by Sun (like Active Server Pages (ASP) by Microsoft). It has the ability to maintain its state after many server transactions. This is done using HTTP cookies, sessions, or Uniform Resource Locator (URL) rewriting.

The Servlet API defines the expected interactions of a Web container and a servlet. A Web container is essentially the component of a Web server that interacts with the servlets. The Web container is responsible for allocation of a URL to a particular servlet and ensures that the URL has the correct access rights.

A servlet is an object that receives requests and generates a response based on the request. The API defines HTTP subclasses of the generic servlet requests and responses as well as an HTTP session object that tracks multiple requests and responses between the Web server and a client. Servlets may be packaged as a Web application.

Web container

A servlet container comprises essentially the component of a Web server that hosts and interacts with JAVA servlets. A servlet container controls the servlets that are deployed within the Web server and is responsible for forwarding the requests and responses for them. It has the functionality of mapping a URL to a particular servlet and of ensuring that the process requesting the URL has the correct access rights.

Some examples of free Web containers:

- Jakarta Tomcat[21]
- Jetty[22]

Some examples for commercial Web containers:

- IPlanet[23]
- Atlanta's ServletExec[24]

Tomcat and IPlanet servers

IPlanet and Tomcat are software programs that are working as a daemon serving Web documents. Daemon is a particular class of computer program that runs in the background of an operating system.

Every Web server program operates by accepting HTTP requests from the network, and providing an HTTP response to the requester. The HTTP response consists typically of an HTML document, but can also be a raw text file, an image, or some other type of document Tomcat Server.

Tomcat functions as a servlet container developed under the Jakarta Project at the Apache Software Foundation. The Tomcat servlet engine often appears in combination with an Apache Web server. Tomcat can also function as an independent Web server in it self. Since its developers, Jakarta Company, write Tomcat in JAVA, it runs on any operating system that has a JVM.

One of the few differences between these two servlet containers is the licensing policy of the developers. Tomcat comes from The Apache Software Foundation working on its open-source software projects. IPlanet is a commercial Web server solution.

21 http://jakarta.apache.org
22 http://jetty.mortbay.org/jetty
23 http://www.iplanet-inc.com/
24 http://www.newatlanta.com

4.4 Output formats

PDF

The Portable Document Format (PDF) is a file format developed by Adobe Systems for representing documents. The usage of this format is independent of the original application software, hardware or operating system used to create those documents.
A PDF file allows describing documents containing any combination of text, graphics, or some images in a resolution and device independent format. These documents can consist of one or thousands of pages, very simple or extremely complex with a rich use of fonts, colors, images or a variety of graphics. This format is an open standard, so anyone is allowed to write application that can read or write PDFs royalty free.

Free readers for many platforms are available for download from the Adobe Website[25].

Technology

PDF is primarily the combination of three technologies:

- a cut-down form of PostScript[26] (PS) for generating the layout and graphics,
- a font-embedding/replacement system to allow fonts to travel with the documents, and
- a structured storage system to bundle these elements into a single file, with data compression where appropriate.

PostScript

PostScript is a computer language – or more precisely a page description language – that is run in an interpreter to generate an image. This process requires a fair amount of resources.

PDF is a subset of those PS language elements that define the graphics, and only requires a very simple interpreter. For instance, flow control commands like `if` and `loop` are removed, while graphics commands such as `lineto` remain.

25 http://www.adobe.com/products/acrobat/readstep2.html
26 PostScript (PS) is a page description language used primarily in the electronic and desktop publishing areas.

That means that the process of turning PDF back into a graphic is a matter of simply reading the description, rather than running a program in the PostScript interpreter. However, the entire PS world in terms of fonts, layout, and measurement remains intact.

As a document format, PDF has several advantages over PostScript. One is that a document resides in a single file, whereas the same document in PostScript may span multiple files (graphics, etc.) and probably occupies more space. In addition, PDF contains already-interpreted results of the PostScript source code, so it is less computation-intensive and faster to open, and there is a more direct correspondence between changes to items in the PDF page description and changes to the resulting appearance of the page. Finally, if displayed with Adobe Reader, a font substitution strategy ensures that it does not matter if the user does not have installed all fonts used in a PDF file – all of them will be inserted in Adobe Reader.

Types of content – two examples

A PDF file for a map is often a combination of *vector graphics*[27] layer, text and raster graphics. For example, the general map of the US[28] (compare Figure 2) uses:

- Vector graphics for coastlines, lakes, rivers, highways, marking of cities and highways symbols. On zooming in, the curves remain sharp; they do not appear to be made up enlarged pixels.
- Text stored as such is scalable. It is also possible to copy the text.
- Raster graphics for showing mountain relief – on zooming in, this consists of enlarged pixels.

Figure 2: Cut-out from general map of the USA

27 Vector graphics or geometric modelling describes the use of geometrical primitives such as points, lines, curves, and polygons to represent images in computer graphics.
28 http://nationalatlas.gov/printable/images/pdf/reference/genref.pdf

An example of a PDF map without raster graphics can be found in the *CIA World Factbook*[29] *map of the Arctic*[30] (Figure 3). The blue color of the sea is not filled neatly up to the vector graphics coast line, but just as greatly enlarged raster graphics, giving a cruder result (noticeable when highly zoomed in).

Figure 3: Cut-out from the CIA World Factbook map of the Arctic

29 http://www.cia.gov/cia/publications/factbook
30 http://www.cia.gov/cia/publications/factbook/reference_maps/pdf/arctic.pdf

5 Open-source licensing policy

One of the tasks of this thesis was to ensure that developed software complies with the IIASA open source policy. Therefore the principles of that policy are described below.

GNU General Public License

The General Public License (GPL) is a free software license, originally written by Richard Stallman for the *GNU project*[31]. It has since become one of the most popular licenses for free software. The latest version of the license, version 2, was released in 1991. The GNU Lesser General Public License (LGPL), another commonly used license, is a modified version of the GPL intended for some software libraries.

The GPL grants the recipients of a computer program the following rights, or "freedoms":

- The freedom to run the program, for any purpose;
- The freedom to study how the program works, and modify it. (Access to the source code is a precondition for this.);
- The freedom to redistribute copies;
- The freedom to improve the program, and release the improvements to the public. (Access to the source code is a precondition for this.);

In contrast, the end-user licenses that come with proprietary software rarely grant the end-user any rights, and even attempt to restrict activities normally permitted by law, such as reverse engineering.

The primary difference between the GPL and more "permissive" free software licenses such as the BSD License is that the GPL seeks to ensure that the above freedoms are preserved in copies and in *derivative works*[32]. It does this using a legal mechanism known as “copy left”, invented by Stallman, which requires derivative works of GPL-licensed programs to be licensed as well under the GPL. In contrast, Berkley Software Distribution (BSD)-style licenses allow for derivative works to be redistributed as proprietary software.

31 A project to create a complete software operating system that is free of charge.
32 Derivative work is an artistic creation that includes aspects of work previously created and protected.

By some measures, the GPL is the single most popular license for free software (also known as open-source software).

Granting of rights

The terms and conditions of the GPL are available to anybody receiving a copy of the GPL'ed work ("the licensee"). Any licensee accepting the terms and conditions is given permission to modify the work, as well as to copy and redistribute the work or any derivative version. The licensee is allowed to charge a fee for this service, or do this free of charge.

The GPL additionally states that a distributor may not impose "further restrictions on the rights granted by the GPL". This forbids, i.e., the distribution of the software under a non-disclosure agreement or contract. . Distributors under the GPL also grant a license for any of their patents practiced by the software, to practice those patents in GPL software.

Copy left

The GPL does not give the licensee unlimited redistribution rights. The right to redistribute is granted only if the licensee includes the source code (or a legally binding offer to provide the source code), including any modifications made. Furthermore, the distributed copies, including the modifications, must also be licensed under the terms of the GPL.

This requirement is known as "copy left", and it gets its legal strength from the fact that the program is copyrighted. Because it is copyrighted, a licensee has no right to modify or redistribute it, except under the terms of the copy left. To exercise rights normally restricted by international copyright law, such as redistribution, the terms of the GPL must be accepted, conversely, if copies of the work are distributed without abiding by the terms of the GPL (for instance, by keeping the source code secret), they can be the original author can sue under international copyright law.

The copy left thus uses copyright law to accomplish the opposite of its usual purpose: instead of imposing restrictions, it grants rights to other people in a way that ensures that the rights cannot subsequently be taken away. This is the reason the GPL has been described as a "copyright hack". It also ensures that unlimited redistribution rights are not granted, should any legal flaw (or "bug") be found in the copy left statement.

Many distributors of GPL'ed programs bundle the source code with the executable files. An alternative method of satisfying the copy left is to provide a written offer to provide the source code on a physical medium (such as a Compact Disc (CD)) upon request. In practice, many GPL'ed programs are distributed over the Internet, and the source code is made available over File Transfer Protocol (FTP). For Internet distribution, this complies with the license.

The “copy left” only applies when a person wants to redistribute the program. Private modified versions may always be made without any obligation to divulge the modifications as long as the modified software is not distributed to anyone else. Note that the “copy left” only applies to the software and not to its output (unless that output is itself a derivative work of the program); for example, a Web portal running a modified GPL content management system is not required to distribute its changes to the underlying software.

GPL as a license policy

The GPL was designed as a license, rather than a contract. The legal distinction between a license and a contract is an important one: contracts are enforceable by contract law, whereas the GPL, as a license, is enforced under the terms of international copyright law. Confusion often arises when people think that the GPL is solely enforceable as a contract, which leads to the wrong opinion that "the GPL is unenforceable because the person never agreed to it”.

The situation is simple: without agree to the terms of the GPL, there is no permission to copy or distribute the software released under it, except through a legal exception to copyright such as fair use or first sale, or by some other agreement with the copyright holder. It does not mean that the rules of the GPL do not apply and that a user may thus use the software in anyway he wishes. The default is the restriction of copyright law, not the anarchy of the public domain.

6 Available solutions for online database information visualization

Within this thesis available visualization methods have been reviewed. This review was a starting point for the selection of technology for our RAINS IMPACT software. Below the most important methods are presented, based on examples (projects) found in the Internet. For each example the main features are discussed, as well as adaptation possibilities for future perspectives of the technology.

6.1 Visualization in SVG

SVG as per W3C is the most modern standard for vector graphics for online visualization, but there are still a low number of online mapping projects on the Internet. This is the new competition to the Macromedia Flash vector graphic environment.

A very good example of this visualization possibility is presented on the Website *www.carto.net*. This is a very good address for all kinds of information about cartography via the Internet. The SVG software produced by *www.carto.net* is called *OpenSVGMapServer* and is available in the version 1.01. This tool is an open-source set of scripts that will dynamically generate a vector map from spatial data in a database when run on a Web server.

Technology of *OpenSVGMapServer* 1.01

The scripts are in the PHP scripting language and are designed for use with a MySQL database. The data generated are in SVG format, with attribute data in the form of ECMAScript / JavaScript arrays. The files can be viewed through the Internet Explorer Web browser with the SVG Viewer (a free browser plug-in from Adobe). Also included in the OpenSVGMapServer software distribution is a script that will export geographic information system (GIS) data and associated formatting information from the desktop GIS software application Geographical Information System (ArcView) into a MySQL format suitable for use with OpenSVGMapServer.

This initial distribution is mainly for testing purposes, as the software is not ready for full use. The most important limitation is that *OpenSVGMapServer* only works with the IE 4.x and 5.x browsers.

Licensing

OpenSVGMapServer is distributed under the terms of the GNU GPL.

Features

- Map features fitting into the current map extension and set to display at the current scale are loaded dynamically along with their attributes on zoom and pan.
- For each session, a log is maintained of all features loaded, and this log is checked for each new request, so that individual features are downloaded only once.
- On zooming, layers and their legend symbols are displayed or hidden based on minimum and maximum scale settings for the layer.
- The new current scale is displayed on zoom.
- X and y coordinates are displayed on mouse move.
- Features can be labelled either automatically on load or on user clicking feature.
- Custom pan and zoom controls are used (does not use Adobe SVG Viewer defaults).
- Map data can be dynamically selected or filtered through a query interface.
- Users can add their own text labels.
- Layers can be switched off and on.
- For unprojected map views (maps in geographic coordinates/decimal degrees of longitude and latitude), point layers can be loaded optionally from an external MySQL database using longitude and latitude fields. This feature enables dynamic mapping of data in the native database.
- ID function: clicking a feature that has attributes brings up an attribute table loaded from local data. Optional linking of features to external files or online databases (e.g., click on a contaminant release site to bring up that site's record in a federal online database).

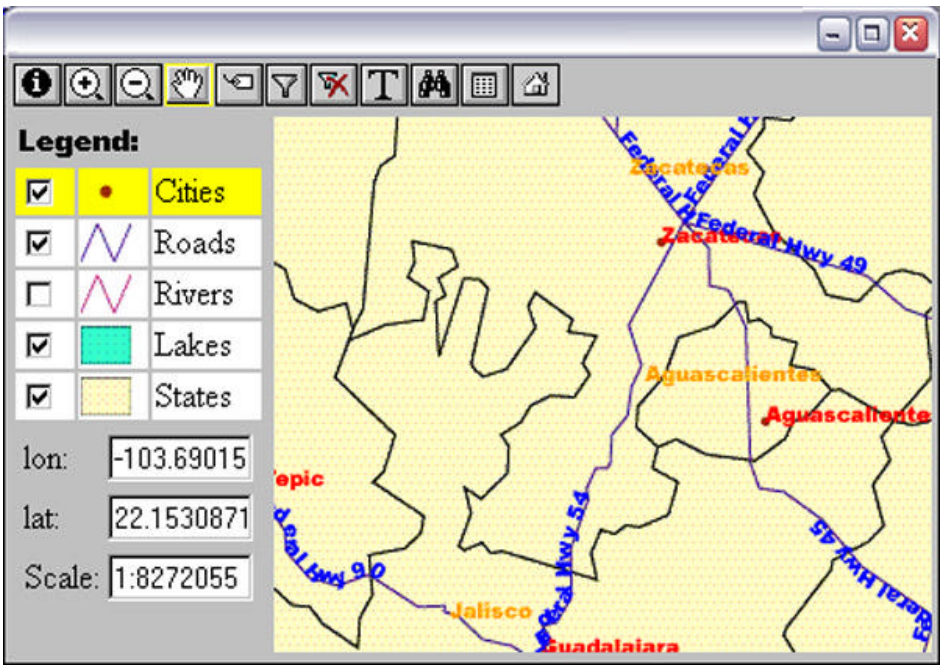

Figure 4: Screen shot from OpenSVGMapServer 1.01

http://www.carto.net/projects/open_svg_mapserver/

Adaptation possibilities

The *OpenSVGMapServer*, as a lot of other SVG online visualization projects on the Internet, is not really adaptable to our mapping model.

The reasons are, on the one hand, dependent on the programming languages – SVG, PHP and the database environment MySQL, and, on the other hand, by the solution described.

PHP do not support JAVA servlets, and without them this project is not adaptable to the mapping model. The most widely used open-source projects on the Internet based on this environment are PHP and MySQL.

The reason also lies in the SVG because this online programming language has a relatively big problem with the plug-ins for the different browsers. During the test phase an attempt was made to try to install SVG plug-ins on the most important browsers (Internet Explorer 5.x, 6.x, Mozilla - Firefox, Netscape and Opera), but any simple SVG animations worked smoothly only on the Internet Explorer. Installations on Mozilla, Netscape and Opera are somewhat difficult, because the user has to copy or delete “some” files in the browser parent directory in the operating system. This is the one of many reasons for the non-popularity of SVG.

There are two developer companies for the SVG plug-ins: Adobe Systems and Corel Corporation. Even Adobe does not like to use their vector graphic solution on their Website[33]. It is strange, but Adobe website includes only Macromedia Flash or Shockwave animations. There are only a few examples of SVG applications but they are separated as their own category.

The deficiency of using the SVG as programming language is very limited support from Adobe or Corel, and a small number of information forums[34], manuals, tutorials, or other types of aid.

Future of SVG projects

Adobe has publicly announced a distribution for the Adobe SVG viewer in excess of $150 million, and that is just one of the many SVG implementations around today. The Adobe SVG viewer is bundled with the Adobe Acrobat Reader, and will be bundled with Real Player in the future.
This is the area in which support for SVG needs to develop, but traditionally it is the area that takes the longest to grow. Also, it should not be forgotten that every text editor, such as Notepad, Emacs or VI, is already an SVG authoring tool.

Adobe wants to create a new SVG open-source project to make this vector graphic programming language more attractive for programmers. But this is not expected to happen in the next year.

According to the words of Dean Jackson[35] (June 6.2002)[36]

> *"The popularity of SVG is growing, and becomes stronger with every new implementation"*

Perhaps this is true but it takes a very long time and it is still, after two years, not there.

33 http://www.adobe.com
34 http://groups.yahoo.com/group/svg-developers
35 Dean Jackson is member of W3C since 2000.
36 http://www.oreillynet.com/pub/a/javascript/2002/06/06/svg_future.html

6.2 Visualization in Macromedia Flash

Macromedia presented Flash for the first time in 1995 – an initial version with basic editing tools. For the past ten years Flash has been the “king” of vector graphics on the Internet and it still is. Flash, because of its scripting language ActionScript, can be used in very different areas of online visualization – as in online mapping.

The most famous example of Flash as an online mapping tool is the French Website *www.geoclip.net*.
Géoclip Flash viewer acts like a GIS software to display the base map. It reads coordinates and transforms them within the coordinate system of the user's screen. It can zoom in and follow mouse movements to move the base map. Each background polygon can be managed separately, for example by changing its colors according to the value of an indicator that is extracted on demand from the database. Flash intersection operators allow objects located in a user defined "buffer" to be selected automatically.

Géoclip Flash viewer draws its flexibility from the separation of three elements: geographic data (coordinates), statistical data (texts and figures), and finally the interface (buttons, dialog boxes). Statistical data is very easily converted from the GIS to a database format that is widely used on Internet servers, such as MySQL, Access, SQL Server, ORACLE, Sybase, Informix, and PostGreSql.

These base maps are used by geographic information system software – MapInfo, Arc View, Géoconcept, etc.

Licensing

Géoclip Flash viewer is commercial software and the provider offers three different categories of prices: “Local”, “Server” and “National” application. They differ from each other in the size, how detailed the maps, in the more or less powerful tools included, and in database connections. The prices range between €8,000 to 24,000.

Features

Géoclip Flash viewer uses a variety of Flash features that allow for interactivity and a smooth, fast, and intuitive interface. An example of a map produced with this technology is presented in Figure 5. Whether by clicking on the map or by using the many buttons available on the tool palette, various interactive features can be used:

- Choosing the area to be mapped
- Zooming and moving
- Viewing additional information
- Selecting geographic areas
- Plotting the statistical distribution for intuitive threshold adjustments
- Setting cartography parameters (selecting variables, colors, thresholds, etc)
- Setting printing parameters

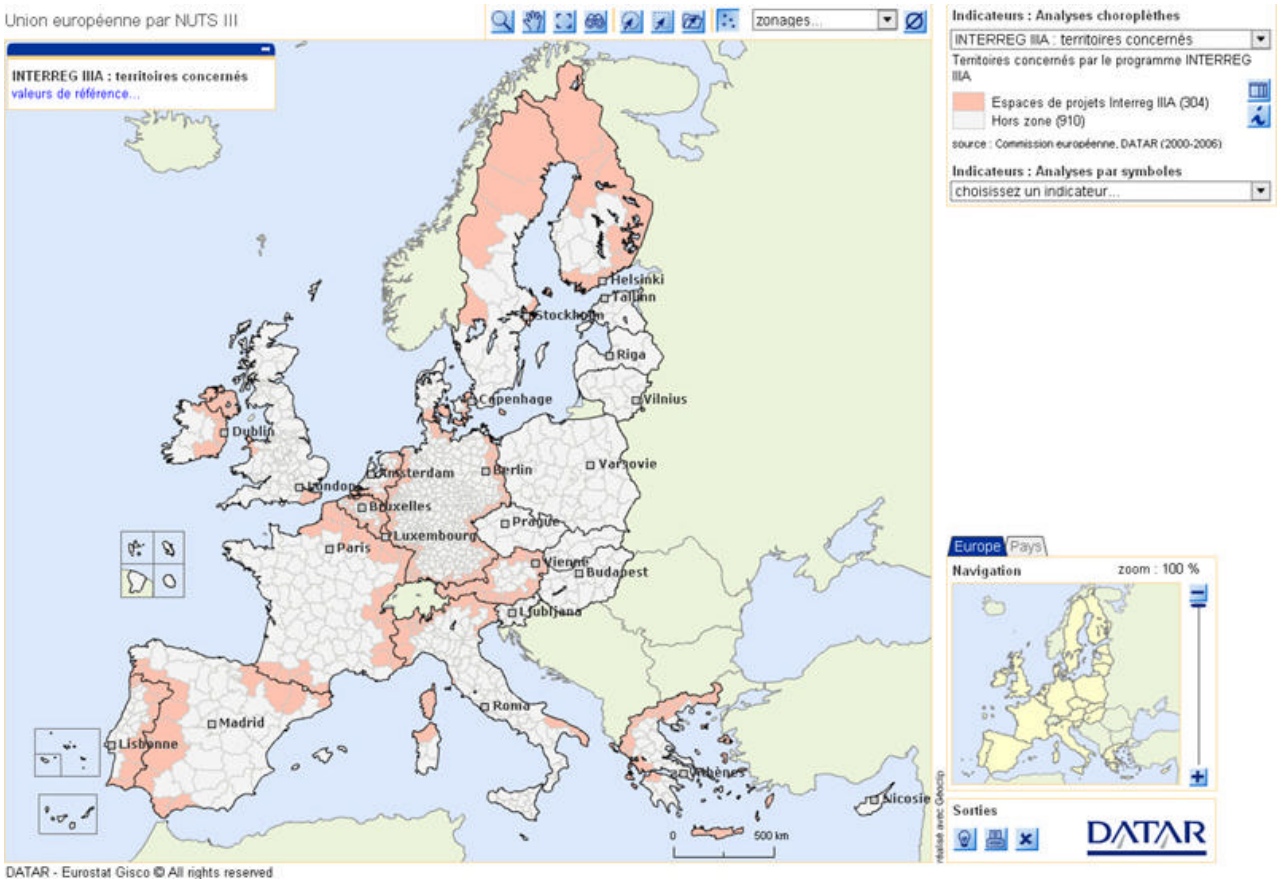

Figure 5: Screen shot of EU map (NUT III) with Géoclip Flash viewer

http://www.geoclip.net/danseuse/geodb_en.php

Adaptation capability

In fact, *Géoclip Flash viewer* is a very attractive and functional solution of online map visualization. Flash player is on more than 95% of all computers and compared for JAVA production of Flash animations is relatively easy.

Géoclip Flash viewer works very well with very simple shapes and with a small number of polygons. However the project presented in this paper will include maps that will be made of up more than 20,000 small polygons. Unfortunately, it is impossible to show this with a Flash viewer. Flash does not have any problems with large homogeneous surfaces, but a great number of polygons presents a huge challenge to any Flash tool. The presentation of Flash files on the screen depends not only on the software - Flash viewer (plug-in) but especially on the speed of the computer (hardware). It could be a "long story" for a user with an old computer.

Producing PDF files from SWF files is also not easy. Either the software producer wants to develop this solution by himself or he has to buy an already programmed SWF-to-PDF converter. These converters are usually not open-source.

The biggest obstacle for using a Flash mapping tool is the licensing side of Flash projects. All the mapping tools that can be found on the Internet are not open-source products. So it is impossible to develop GPL software from a commercial product.

The future of Flash projects

When somebody thinks about Flash, he is probably thinking about a client-side development environment and client-side development mentality. Macromedia is starting to develop a new model: The Royal Model. This new model will result in some difference to the past. Macromedia wants to organize the development on the server, and deliver it down to the client. It is a very similar model to JSP or ASP or any other of scripting languages. The Royal Model is not actually designed to create Flash movies but for creating Flash interfaces.
Macromedia wants to offer a more efficient programming language. ActionScript, which is already included in the latest version of Flash (Flash MX), allows creating a lot of client-server interactivity.

In the future Macromedia also wants to allow some other W3C programming language standards, like the already very popular XML.

In an online *Community MX*[37] interview[38] Mr. Lucian Beebe, a senior product manager at Macromedia, was asked a very interesting question by the journalist:

> ***Interviewer**: "What's Macromedia's position on open-source Flash projects?"*
>
> ***Beebe L.**: "We do not really take a position on it. We are not against it, or for it, necessarily. I think there's a lot of good that can come of it. The downside is that we're not in control of it. Not from a monetary perspective, but from a quality perspective. Things can happen to make the quality a disaster, and we have no control over that. Apart from that, we do not say there's anything wrong with it. Community innovations have always been the driver for Flash."*

According to this statement, the software developers can spend a long time waiting for any open-source projects with Flash.

6.3 Visualization in JAVA technology

JAVA technology was created as a programming tool in a small, closed-door project initiated by Patrick Naughton, Mike Sheridan and James Gosling of Sun in 1991.

The JAVA platform is based on the power of networks and the idea that the same software should run on many different kinds of computers, consumer gadgets, and other devices. Since its initial commercial release in 1995, JAVA technology has grown in popularity and usage because of its true portability. The JAVA platform allows you to run the same JAVA application on lots of different kinds of computers.

One of the most famous examples of JAVA technology as online mapping tool is the Australian Website *www.timemap.net*.
ALOV Map, a tool presented on this Website, is a portable JAVA application for publication of vector and raster maps to the Internet. It supports complex rendering architecture and unlimited navigation. It allows working with multiple layers, thematic maps, and hyperlinked features and attributing data.

TMJAVA is a joint project of *ALOV Software*[39]*e* and the *Archaeological Computing Laboratory*[40], *University of Sydney*[41], Australia. ALOV Map was started in November

37 http://www.communitymx.com
38 http://www.communitymx.com/content/article.cfm?cid=A326E0BFDB5DFD79&print=true

2001 as an offshoot from work on the Windows version of TimeMap® software (commenced 1999).

TMJAVA provides many new features including temporal filtering, advanced layer control, rendering of TimeMap MapSpaces, and access to a central index of datasets developed for the Electronic Cultural Atlas Initiative *www.ecai.org*[42].

There are two approaches for Web mapping with ALOV Map - standalone and client/server.

- The standalone version is the easiest way to publish any GIS data on the Internet. There is no special system or programming experience required to set up a Web GIS. No dedicated server-side is needed.
- The client-server version is more flexible and allows the incremental downloading of data. The server spreads vector maps to client in the most efficient way and uses advanced caching to reduce network traffic. The map features are stored on a SQL database according to OpenGIS SFS. Multi Resolution Seamless Image Database (MrSID) image server and any OpenGIS World Map Service can be used as raster maps source. The server side is based on JAVA servlets and can be integrated into any Web server.

Licensing

TMJAVA is available at no cost for personal, educational and other non commercial purposes. Commercial businesses have to obtain a license from the *University of Sydney*[43].

Features

TMJAVA offers the most important and fundamental features for an online mapping tool. This includes the following possibilities for the user:

- Choosing the area to be mapped
- Zooming and moving the map

---

39 http://alovsoft.com
40 http://www.archaeology.usyd.edu.au/acl
41 http://www.arts.usyd.edu.au/departs/archaeology
42 http://www.ecai.org
43 http://www.arts.usyd.edu.au

- Scrolling of any area (whole map or any selected or zoomed area)
- Viewing additional information
- Selecting geographic areas
- Well organized legend
- Protocol of loaded data from the server
- Searching of certain polygons
- A tool to identify the distance between two or more points on the map
- Linking of certain polygons to some Internet addresses (i.e., for detailed information of user selected polygons)

An example of a map produced with *Alov TMJAVA* is shown in Figure 6.

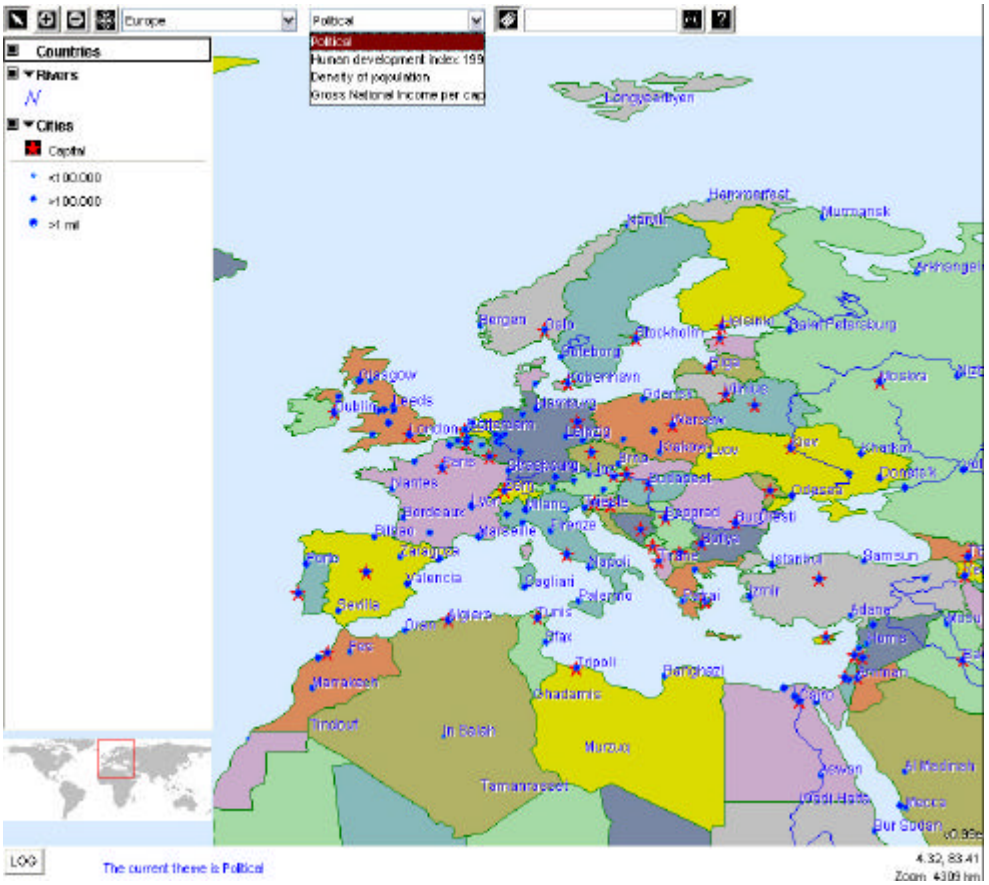

Figure 6: Screen shot of EU map with Alov TMJAVA

http://alov.org/sample/world.html

Adaptation capability

TMJAVA is a very practical und acceptable mapping tool for the IIASA project. The primacy reason for this is the technology. To run a JAVA application on a client computer, the operating system needs JVM, which is, in relation to the plug-in statistic mentioned in chapter 3.3, a very widespread application system. JVM is installed on more than 95% of computers. JVM has any problems with presentation the of big numbers of polygons (in contrast to Flash).

The next positive fact is the development environment. *TMJAVA* runs as a JAVA servlet on the server and as a JAVA Applet on the client side. The databases supported are MySQL, Interbase, MS SQL, Access, SAP DB, Informix, Hypersonic, and PostGreSql. Our mapping model at IIASA is based on the ORACLE database. The difference between ORACLE and PostGreSql is, basically only the license policy (ORACLE is a commercial and PostGreSql a free database system).

The licensing policy of *TMJAVA* also corresponds to the project's requirements. IIASA wants to develop an open-source online visualization tool for its air pollution simulation models and Alov pursues this policy, too. During the next two years, Alov is planning to establish the GNU policy, but present to get the source code Alov requires membership in their consortium.

TMJAVA was optimized to minimize the traffic of data in the client-server communication. It means that the client side gets all map information in the beginning of the transaction, and any zooming or moving of the map does not need any additional data traffic.

Future of JAVA

JAVA is one of the oldest online programming languages. This means that because of the popularity of this development environment there exist a large number of JAVA experts all over the whole world.

JAVA will be on an equal basis with by C these days, this means not as good as it might be, but much better than it could have been. The JAVA development environment has wide support from libraries, platforms and tools, and this ensures the future of JAVA as a programming language. JAVA is also suitable for a large number of applications. It is not always the most appropriate tool, but is used anyhow because it is a "safe" decision.

Programmers need not be “brilliant” to work effectively, which makes JAVA more and more popular.

All these features mean that also in the future JAVA is likely to remain a a very efficient and most popular programming language.

7 Selection of technology for RAINS IMPACT

The criteria analyzed in order to select the technology for RAINS IMPACT is decrypted below. The criteria used were:

- licensing policy;
- difficulty of adaptation to the existing model;
- flexibility of the source code;
- traffic economy;
- availability of plug-ins;
- perspectives for future development;

A summary of the analysis is presented in Table 1.

In the market economy one of the most important criterions for a decision is the financial side of any project. Thus the first criterion was the **licensing policy** of the different theoretically acceptable projects. The open-source projects, so-called no-cost-projects, are mostly applicable in SVG and JAVA. Flash is usually not an open-source development environment.

Furthermore, the scope and difficulty of **adaptation** of the new source code **to the already existing model** was an important selection criterion. Use of SVG or Flash would require complex integration of the new system into the existing one. Because the existing RAINS Web is developed in JAVA, JAVA offers the best alternative, which minimizes the complexity.

Flexibility of the source code was the next criterion. Flexibility means the adaptability to implement new applications even when difficult. Again, because RAINS Web already uses JAVA, this category goes to JAVA. The source code of SVG and Flash projects is, of course, also flexible, but this does not apply to the model.

Traffic economy is well, was considered only using JAVA. Flash has problems with the transfer of large number of polygons, and in the case of SVG practical experiences are not sufficient.

Browser independency, which means the **availability of plug-ins**, is the next point for consideration. Presented in Section 3.2. survey of current users provides a clear and simple answer: Flash and JAVA have the most widely distributed plug-ins in the Web

browser scene. SVG is last in the rank, which increases a risk of failure if this technology is used.

The **perspectives for future development** of programming languages described was the last point for selecting the best environment. 3WC promises an optimistic future for SVG, but this was advertised already two years ago and nothing particular happened. The future of Flash and JAVA is not only guaranteed in the entertainment sector of the Web, but also in client-server applications.

Technology	License policy (GPL)	Adaptation to existing model	Flexibility of source code	Traffic economy	Plug-in distribution	Perspectives for future development
SVG	✓	✓	✓	n a	✓	✓
Flash		✓	✓		✓✓✓	✓✓✓
JAVA	✓✓	✓✓✓	✓✓	✓✓✓	✓✓	✓✓✓

Table 1: Technology selection criteria for RAINS IMPACT

Based on the evaluation presented in Table 1, the decision to use the JAVA technology as the most satisfying solution was made.

Furthermore, a decision has to be made how to generate the output files. First of all, the output format had to be selected. Out of all possible formats, the PDF format was chosen. This is justified by further trouble-free handling with this format. It can be smoothly converted to other vector graphic formats (i.e., EPS) and the distribution of the PDF viewer is very sophisticated (compare to Sector 4.4).

The creation of a PDF vector graphic file from a JAVA Applet window requires a JAVA library that allows generating PDF files. One possibility the in-house development (programming) of such a library. However, such a solution is time-and resource consuming. Besides, this would be like "reinventing the wheel", since there are existing libraries offering satisfactory solutions. Commercial and non-commercial (open-source) solutions were found on the Internet. An example of a commercial product is *ICEpdf*[44]. There exists also an open-source tool called *iText*[45]. The commercial solution has two big disadvantages: high cost and no flexibility in the source-code. *iText* offers source-code, support and a relatively extended users' community. These were the reasons why

44 http://www.icesoft.com/products/icepdf.html
45 http://www.lowagie.com/iText/

it was decided to take the non-commercial solution. The *iText* classes are very useful for users who need to generate read-only, platform independent documents containing text, lists, tables and images. The library is especially useful in combination with JAVA technology-based Servlets.

Further reason above, the decision was made that the *iText* PDF creation library is most suitable for our project because it based on JAVA Servlet technology and follows the open-source policy.

8 RAINS IMPACT visualization software

This chapter describes the software implementation and demonstrates examples of output. First, the general architecture of the system is discussed, presenting the sequence of steps in which calculations are done and provide basic information about adopted solutions with regard to client-server conversation. Next, some basic information about the original product *TMJAVA* which was a starting point for the work is given. In the further section the changes (on the server and the client side) we have implemented to assure compatibility with the already existing RAINS Web model are discussed. For adapting the *TMJAVA* to the already existing RAINS Web model we had to define a client-server communication protocol. New applications needed to be added in order to ensure the full functionality of the model, which is described in a later part of the chapter. The next two sections are devoted to the description of solutions adopted for storing the output (as a PDF files), and handling administrative tasks. Finally, at the end of the chapter model output through demonstrating examples of maps calculated and visualize d with RAINS IMPACT are illustrated.

8.1 General architecture

Based on the overview of currently available tools presented in the previous section the decision was made that the visualization software should be executed in the following main steps:

- Client calls up a graphic JAVA Applet template;
- Using the menu from the graphic (sub) window of the browser, the user has the possibility to change settings of the graph or request new data from the servlet.
- JAVA Applet script sends a request to the servlet.
- New data is processed by an appropriate script. The data is sent to the browser where the script updates the graphic according to the settings specified by the user.
- To complete this task, the template should call the servlet to deliver data. Graphic generation should be done on the client Web browser.

Such an architecture means that visualization of the maps takes place in an interactive way (“on the fly”), Thus the maps will be not saved on the server – these will be produced only by user demand. This requires that there be become relatively small data

traffic between the client and server side. A schematic representation of "client-server-discussion" that meets that requirement is shown in Figure 1.

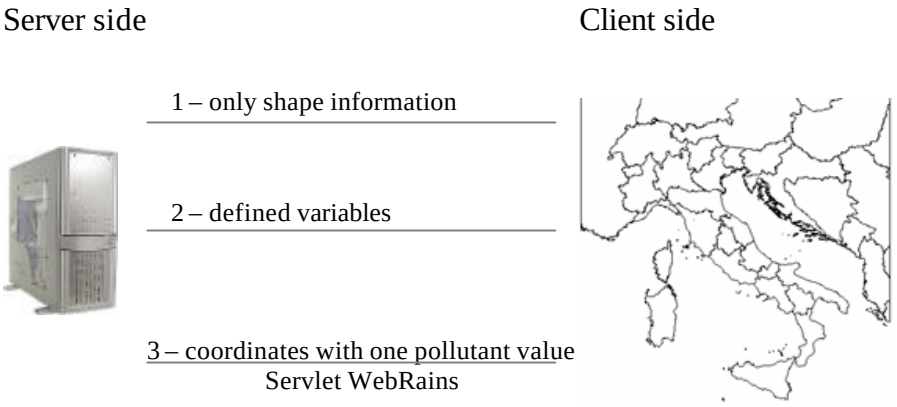

Figure 7: General visualization of client-server-conversation

At the beginning of the client-server-conversation, the server sends only the shape information of the map to the client. It means that the client will get only the contour of the map (administration borders). The client represents this information as outlines in the browser window. After clicking a button, i.e., "simulate new scenario", which means "please send me detailed information for a specified pollution scenario", the client sends the set of variables he wants to be included to the server (as an HTML code). Based on this, the server prepares a logical SQL-query statement, and sends it to the ORACLE database, gets the information back from the database, and forwards it to the client. Information that is sent to the client is only a value of the indicator plus coordinates of a grid for which this value was calculated. Since the number of the coordinates could be very huge (the European map has 20.000 grids) it is very important that the data transmission includes only coordinating points and the value of an indicator.

8.2 TMJAVA development environment

This project we used as a starting point a java development tool *Apache Ant* which was used by the *TMJAVA* developer from the University Sydney. Ant is a JAVA-based builder tool. In theory, it is similar *make*, without *make*'s wrinkles. Because the experience with this development environment was not sufficient and in the past the *JBuilder* from Borland was used, the decision was made to change this part of the tool and to use JBuilder instead. *JBuilder* had many advantages, such as graphical user interface, very easy installation, very easy debugging of servlets, and complete support from Borland.

The disadvantage of *JBuilder* is the commercialization of this software. The licenses are very expensive, but two licenses of *JBuilder 2005* with *Tomcat 5.x* as servlet container were available. To develop the *TMJAVA* in *JBuilder*, changes described in Section 8.2.2. had to be made

8.2.1 Technology solutions used

General

TMJAVA needs, according to the already existing RAINS Web model, two components to be installed: database and servlet container. TMJAVA was developed, in fact, for using the MySQL database but the change it to the ORACLE data system was made (see Section 8.2.2). On the Web server side the servlet container *iPlanet* was used. A smooth handling on a standalone PC is important. The PC should have a complete database installation. To achieve this goal a second, non-commercial servlet container, *Tomcat* from Apache is used.

Server side architecture

The structure of the database is explained below. The description refers to the original database of the *TMJAVA*, which is called *Clearinghouse*. *Clearinghouse* is an SQL database or an XML document including records where the description of GIS datasets is stored. These records include descriptive data for resource discovery allowing identification of resources relevant to a particular place or thematic interest. They also contain connection parameters for each dataset which allows the server to connect to the data servers across the Internet (or to local files).

TMJAVA needs a very significant XML file to start the client server connection. This file contains the information for the main servlet called *pump*, about the location itself, connection of the database (server name, port number, user name and password, driver and server type), and administrator name and password. The major problem was to ensure the automatic finding of the location of this control-XML-file in the operating system. The file is called *mapserv-home.xml*. A satisfactory solution has now been found.

A short source code of *mapserv-home.xml* is presented below:

```
01 <?xml version="1.0" encoding="ISO-8859-1"?>
02 <mapserv>
03      <clearinghouse type="db">
```

```
04          <xml url="/t/rains/mapserv-home/mapserv.xml"/>
05          <database user="rains_user" password="rains_password"
06           url="jdbc:oracle:thin:@yangtze.iiasa.ac.at:1521:resrch2"
07           driver="oracle.jdbc.driver.ORACLEDriver" server="ORACLE"/>
08       </clearinghouse>
09     <master user="someuser" password="somepassword" ip="*"/>
10 </mapserv>
```

In the first line is the definition of the XML language standard. *Clearinghouse* describes the type of the data sources (db= database and file=XML files). The lines 04-07 describe the connection to the database. The line 09 includes the administrator name and his password.

Database structure

The *Clearinghouse* database consists of six main tables: TM_BLOBS, TM_COUNTER, TM_DATASETS, TM_INSTANSE, TM_SERVERS and TM_USER.

TM_BLOBS

This table contains the information about the datasets projects (i.e., current number of data-set-entries, the auto incremental number of the project, the kind of project and information about the design of the legend).

	TMB_ID	TMB_DS_ID	TMB_EL_ID	TMB_VALUE
1	90	222	16	<symbol val="500.00" fill="117:17:23" outlined="no" label="500.00"/>
2	87	217	16	<?xml version="1.0" encoding="ISO-8859-1"?> <project zmin="50" usetime="no" zoomunits="km" backcolor="255:255:255"> <domain ymax="440000" xmax="95000" name="Zeeland" ymin="355000" xmin="8000"/> <map index="m1" name="NL Simulation"/> <layer password="yes" label="Simulation" visible="yes"> <dataset full="yes" id="212"/> <renderer type="gradcolor" field="VALUE" map="m1"> <symbol val="0" style="1" fill="212:239:252" outlined="no" label="0" /> <symbol val="230.00" style="1" fill="0:169:157" outlined="no" label="230.00" /> <symbol val="460.00" style="1" fill="166:206:57" outlined="no" label="460.00" /> <symbol val="690.00" style="1" fill="255:223:79" outlined="no" label="690.00" /> <symbol val="920.00" style="1" fill="242:101:34" outlined="no" label="920.00" /> <symbol val="1150.00" style="1" fill="117:17:23" outlined="no" label="1150.00" /> <symbol val="1380.00" style="1" fill="41:0:3" outlined="no" label="1380.00" /> </renderer> </layer> <layer password="yes" label="NL Regions" visible="yes"> <dataset full="yes" id="130"/> <symbol outline="0:0:0" filled="no"/> </layer> <layer password="yes" label="NL Cover" visible="yes"> <dataset full="yes" id="132"/> <symbol style="1" outline="0:0:0" filled="no" /> </layer> <ECAI_ID>217</ECAI_ID> </project>
3	89	219	16	<symbol label="800.00" outline="41:0:3" fill="41:0:3" val="800.00"/>
4	88	218	16	<symbol val="12.00" fill="41:0:3" outlined="no" label="12.00"/>
5	64	172	16	<symbol val="4.00" style="1" fill="117:17:23" outlined="no" label="4.00" />

Figure 8: Screen shot of the table TM_BLOBS

TM_COUNTER

This table is a simple alternative for storage of next numbers of different devices (i.e., the number of all datasets, of instances, all projects, and users).

	TABLE_NAME	NEXT_ID
1	TM_DATASETS	272
2	TM_INSTANCE	7047
3	TM_BLOBS	125
4	TM_USERS	1
5		

Figure 9: Screen shot of the table TM_COUNTER

TM_DATASETS

This table includes all information required about all stored maps (i.e., the identification number, the user who stored this in the database, date of storage and update to the database, title, status, that is if the dataset is public or not, four size-describing-points of the saved map, etc.)

	DS_ID	DS_USR_ID	DS_REGDATE	DS_UPDATED	DS_AUTHRSE	DS_TITLE	DS_STATUS	DS_XMIN	DS_YMIN	DS_XMAX	DS_YM
1	130	0	2005-02-01 00:00:0(	2005-02-01 00:00:0(		NL_5x5_prl_polyline	6	13455.2998047	306891.40625	277896.1875	6
2	131	0	2005-02-01 00:00:0(	2005-02-01 00:00:0(		nl_j5x5_2_region	6	13455.2998047	306891.40625	277896.1875	6
3	132	0	2005-02-01 00:00:0(	2005-02-01 00:00:0(		cover	6	-8267.4003906	292435.125	326867.34375	
4	173	0	2005-03-03 00:00:0(	2005-03-03 00:00:0(		EU_regions	6	2358096.5	-3790369.5	8589565	

Figure 10: Screen shot of the table TM_DATASETS

TM_INSTANCE

This table describes what has happened during every user interaction (i.e., the current number of interactions, which dataset or project has been changed, what has been done and some descriptions).

	MD_ID	MD_DS_ID	MD_EL_ID	MD_SCH_ID	MD_STR_VAL	MD_LANG
1	3810	130	4	0	13455.3	
2	3811	130	5	0	277896.2	
3	3812	130	6	0	306891.4	
4	3813	130	7	0	618814.7	
5	3814	130	8	0	NL_5x5_prl_polyline	
6	3815	130	36	0	0	
7	3816	130	41	0	rafa	
8	3817	130	43	0	NL_5x5_prl_polyline	
9	3818	130	49	0	NL_5x5_prl_polyline	
10	3819	130	80	0	PRVNR PRVNM IA	

Figure 11: Screen shot of the table TM_INSTANCE

TM_SERVERS

This table includes information for supported (and not supported) JDBC driver classes.

	SRV_ID	SRV_NAME	SRV_JCLASS	SRV_SQL	SRV_INCRMT
1	1	MySQL	org.gjt.mm.mysql.Driver		
2	2	Interbase	interbase.interclient.Driver		
3	3	Sybase			
4	4	MS SQL			
5	5	Oracle	oracle.jdbc.driver.OracleDriver		
6	6	Informix	com.informix.jdbc.IfxDriver		
7	7	DB2			
8	30	MS Access	sun.jdbc.odbc.JdbcOdbcDriver		
9	31	Hypersonic	org.hsqldb.jdbcDriver		
10	32	SAP DB	com.sap.dbtech.jdbc.DriverSapDB		
11	33	PostgreSQL	org.postgresql.Driver		
12					

Figure 12: Screen shot of the table TM_SERVERS

TM_USER

This table includes information about all registered users (i.e., user name, password, user rights, user description – email, address, etc.)

	USR_ID	USR_NAME	USR_USRNAM	USR_PWD	USR_EMAIL	USR_ORG	USR_CNTRY
1							

Figure 13: Screen shot of the table TM_USERS

Servlet container structure

As mentioned before *TMJAVA* is developed in JAVA environment. The structure of the source code is as following: all classes are structured in nine packages, which are presented in Figure 14. A package is a group of related classes and interfaces The main control package is called *org.alov.serv* which includes the main servlet *UploadServlet* (called *pump* – the name is given because of the functionality of this servlet; it "pumps", i.e., transports all information from server to client and back). For the detailed description of all packages please visit *http://org.alov* or see the documentation on the included CD.

Figure 14: Packages of TMJAVA

Client side architecture

Applet structure

TMJAVA uses a JAVA Applet for visualization of any maps on the client side. This application is a very variable solution. Layout of this Applet is defined as an XML file stored on in the root directory of the *Alov Applet*, which can be changed any time.

The layout file is an XML file that specifies how components are placed on the Applet. It defines colors, captions and other properties of applet components. Using a layout file (see Sector 8.6.), the system administrator may add or delete any components. In addition, the layout file may contain resource strings (for localization of messages, tips and captions).

8.2.2 Changes implemented

During the implementation of *TMJAVA* source code to the development environment (*JBuilder 2005*) some classes had to be changed to public ones. Insignificant changes in the structure of some servlets were also made (these will not be explained because it is a very large topic in itself). Many small modifications were also made because of the Windows XP and Solaris X operating systems.

After the JBuilder implementation, the compatibility with our existing model RAINS Web had to be ensured. This means that *TMJAVA* should be able to connect to the database system, which is ORACLE. This task includes the implementing of the ORACLE JDBC, definition of the database URL, and changing the structure of the *clearinghouse* database (see chapter 8.2.1).

Another important modification was the replacing of some data types with others. For instance, since *TMJAVA* used data type *long raw*, and this format is not supported anymore by ORACLE Company it had to be changed to the data type *blob*.

Another very significant change is the inclusion of the new packages *org.alov.user_interface* and *org.alov.itext*. The first package includes classes, which allow the registered and already logged in (in the RAINS Web model) users to make some user specified tasks (i.e., adding a new or changing the legend, editing of scenar-

ios, etc.). *iText* package allows the producing of PDF files from the visualized map (compare Sector 8.5.).

The next very important change was the design of Alov Applet. It should be artless, very simple to understand for every user and similar to the already existing RAINS Web model. The RAINS IMPACT – XML layout file, which is implemented on the CD, creates the following design for Applet (Figure 15):

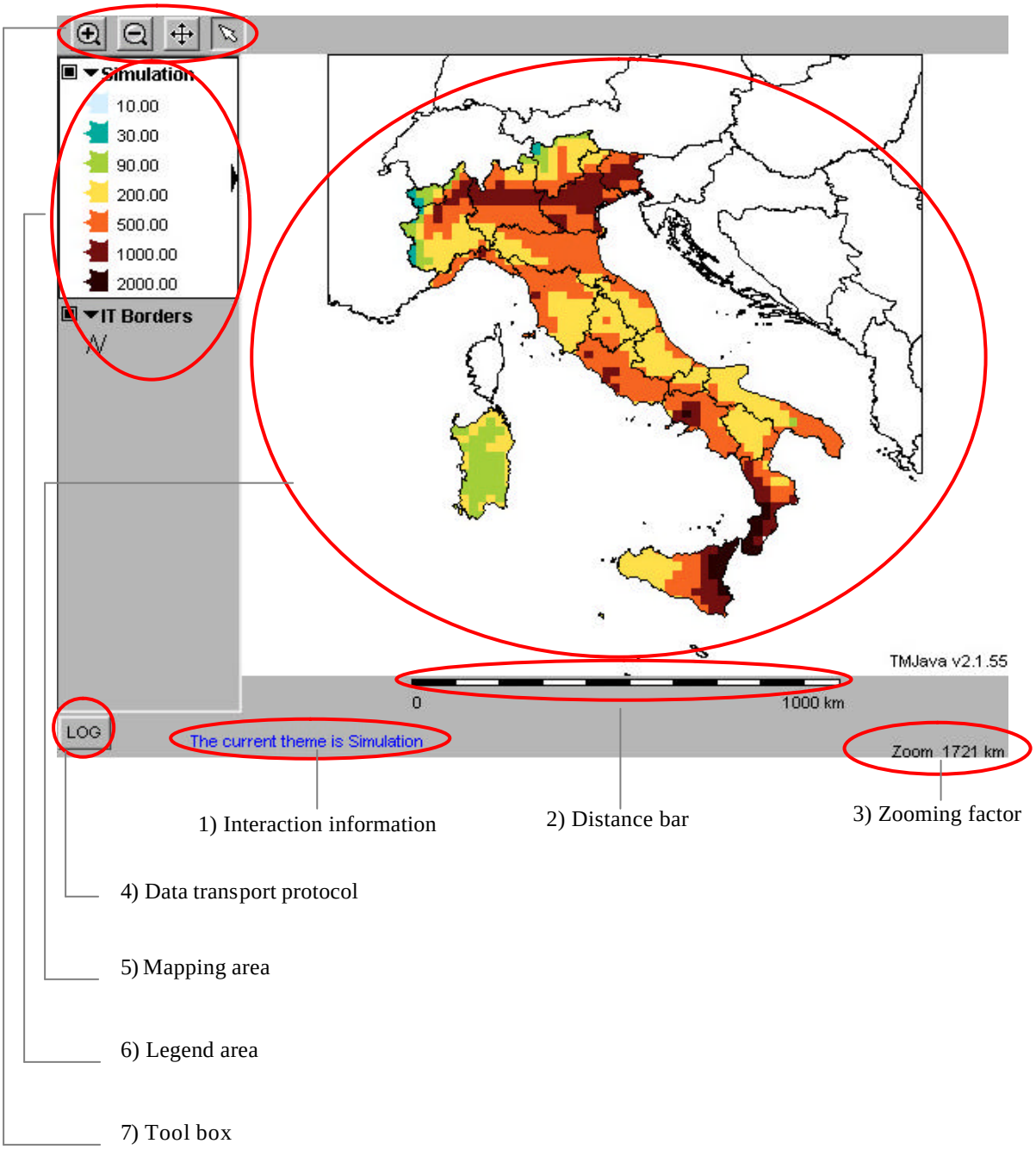

Figure 15: RAINS IMP Applet design

The toolbox (Figure 15, 7) offers the user important tools to interact with the map, such as zooming in or out and scrolling. The selection tool is the default mouse pointer. The legend part of the RAINS IMPACT Applet (Figure 15, 6) offers information about the loaded layers. The "Simulation" layer defines colors and ranges for the visualized map of the user specified scenario (see chapter 7.6.2). The "IT Border" layer contains the administrative borders in the mapping area (Figure 15, 5). For better user orientation inside the RAINS IMPACT Applet a distance bar tool has been implemented (Figure 15, 2), which visualizes the scale of zooming and according to this the zooming factor as a number (Figure 15, 3). The user can also look at the contents of the log file (Figure 15, 4), which gives the information about the sources of the loaded components (maps, legends, etc.), their database-oriented names, loading time and some other (advanced) control values.
Also included is a short informational communication, which tells what is just happened after a interaction by a user (Figure 15, 1).

Build-up of the client side

The new developed client side consists of HTML code that includes four components that are visualized on the Figure 16 as color boxes with numbers on the right top side. The first component (1) is an inline frame which allows a selection of scenario data to be used (definition of version (owner), scenario name and the year). For documentation and explanation of this variables see *http://www.iiasa.ac.at/Web-apps/tap/RainsWeb*. The next component (2) of the HTML code is a JavaScript that constructs three combo boxes (drop-down menu) allowing a selection of the type of indicator to be displayed (MetModel), legend file name and groups of emission sources (regions) to be included in the simulation. During the user interaction, the button "Calculate map query" starts the process of the calculation of appropriate indicators and finally produces the map. RAINS IMPACT Applet (3) is the visualization part of the client side. It allows the user to interact with an already printed map. The last component (4) of the HTML code is the debugging Applet which gives the information about the current status of the calculation (selections, data transfer etc.). This Applet is mostly used for debugging of client side.

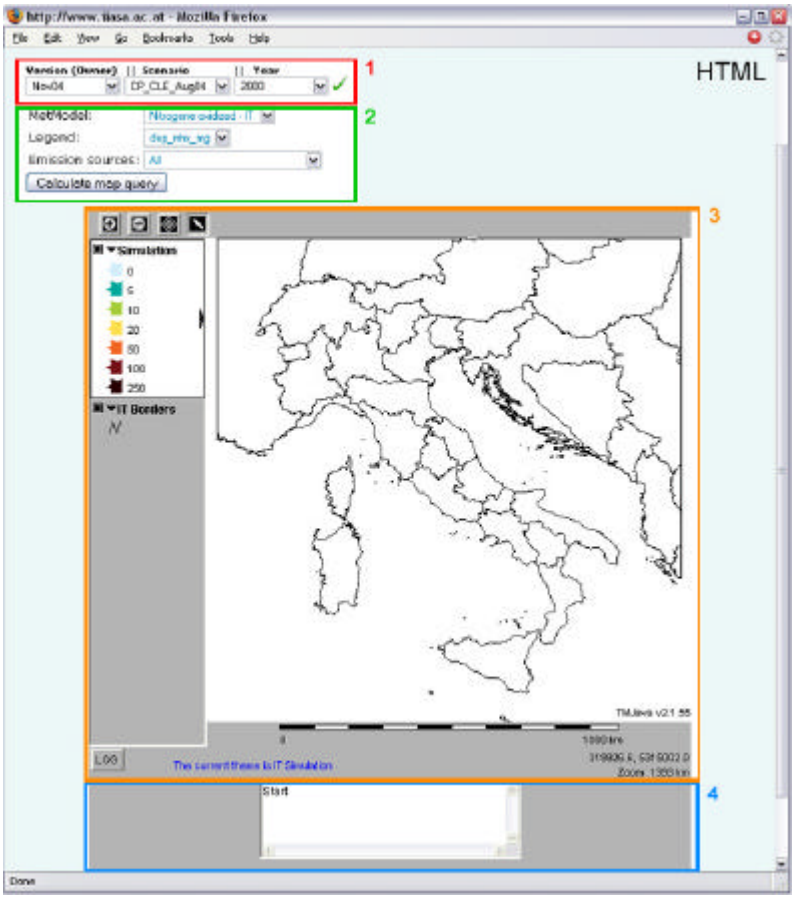

Figure 16: Graphical build-up of the client-side

Build-up of the server side

The built-up of the server side already existed in the RAINS Web. It has been taken over without making important changes. This was one of the requirements for the visualization software.

The server side consists of four components. As Servlet container the *iPlanet* (compare Sector 8.2.1) was used and also alternatively *Tomcat 5.x*. This part of the server side contains all Servlets and JavaScripts. Included the servlet *WebRains* is the already existing online RAINS model and the servlet *rainsmap* includes all classes of the new visualization software RAINS IMP.

ORACLE 10.i is the database system used, but because of the intention to ensure the GPL policy the visualization software for PostGreSql was also developed. This is a non commercial edition of ORACLE.

The connection between the servlet Container and the database ensure the JAVA Data Base Connector (JDBC).

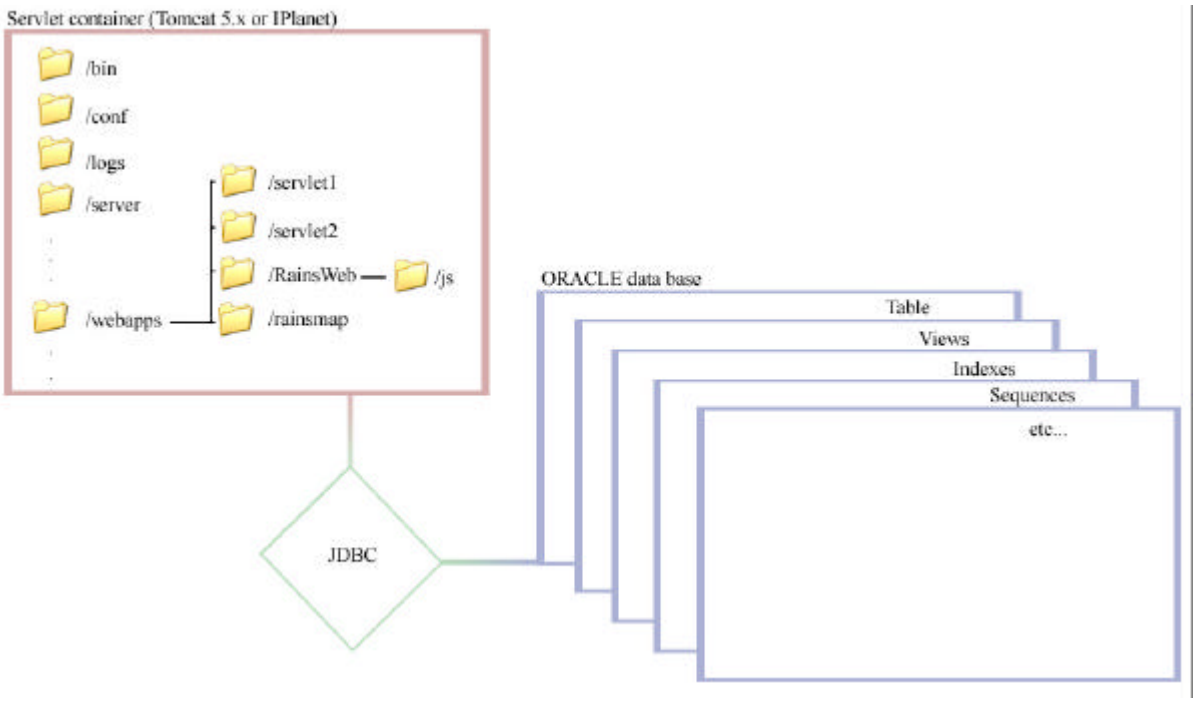

Figure 17: Graphical build-up of the side-side

8.3 Client-server communication

Build-up of the client side as interaction between server and client

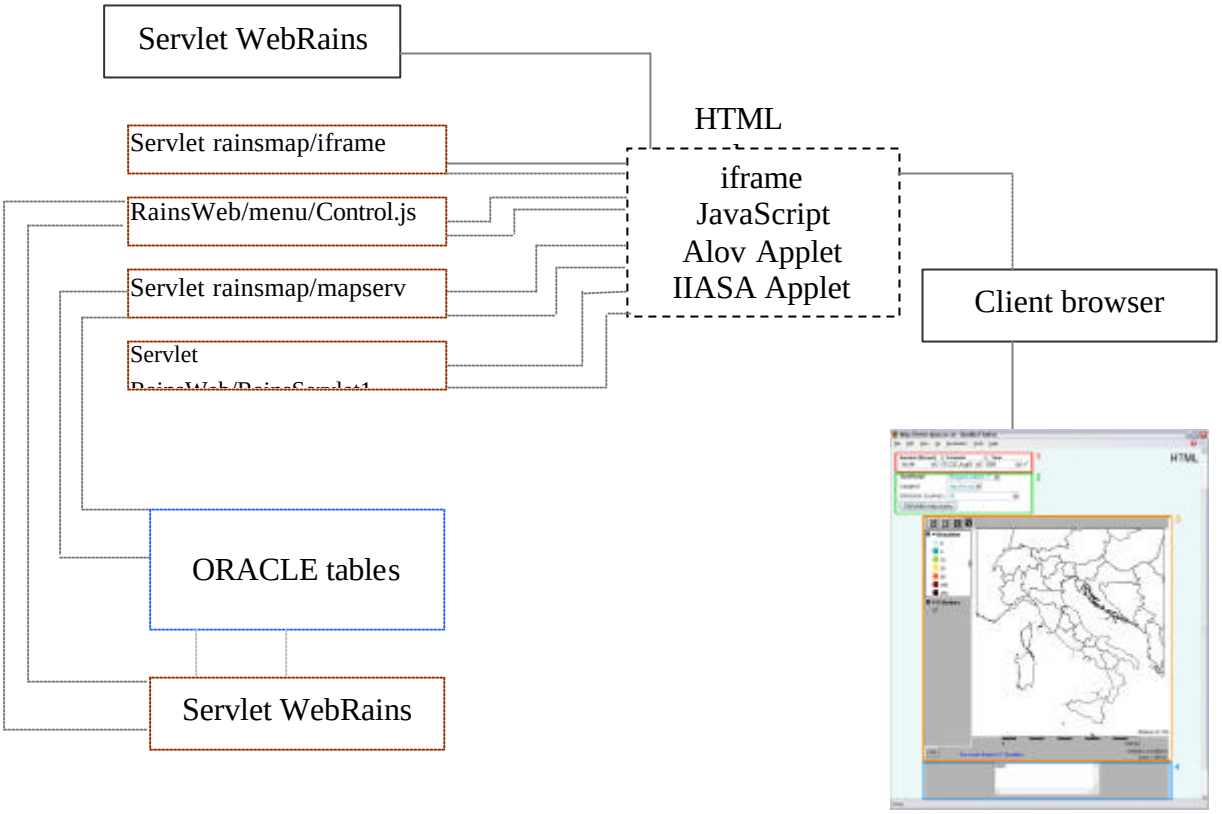

Figure 18: Technical build-up of the client side

At the beginning of the client-server communication, the server sends the HTML code to the client browser. This HTML code contains the following components: an iframe, JavaScript, and two JAVA Applets.

JavaScript is taken from the directory */WebRains/js* and offers the possibility to select parameters to be chosen for the production of the map. To these parameters belong: the type of indicator displayed (e.g., deposition of oxidized nitrogen), legend, type (i.e., ranges of values to be displayed) and emission sources to be included in the map. Selection occurs in three combo boxes. To collect all selection possibilities, it calls up the *WebRains* servlet, which makes a query to the ORACLE database; and after getting a positive answer sends all selected rows to the JavaScript which builds the combo boxes.

The Alov Applet is called up with the following parameter `<param name='pid' value='258'>`. Servlet *mapserv* will search project 258 in the database and send back all selected rows to the Alov Applet, which builds up the legend and the map (outlines only).

The IIASA Applet will be called from a servlet placed in */RainsWeb* called RainsServlet1.

After all parameters are collected, the client browser interprets the HTML code and presents it to the user.

Visualization of the newly selected scenario

If the user wants to visualize a new simulation scenario, he has to define the following three variables:

- Version of model
- Scenario and
- Simulation year

These variables are placed in the iframe.

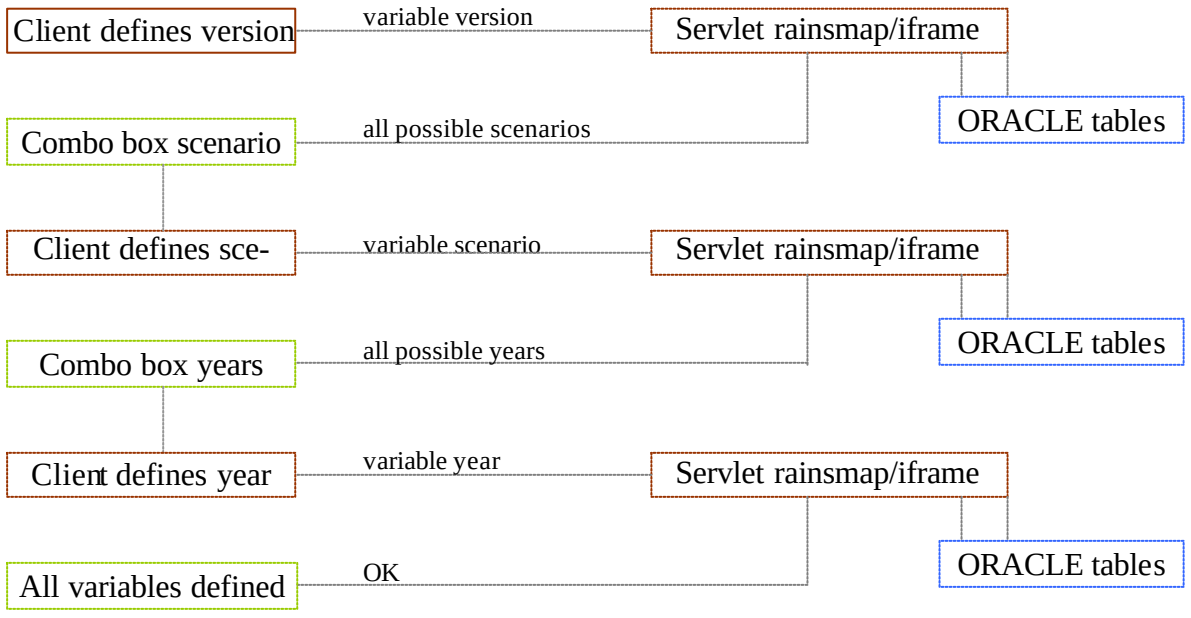

Figure 19: Client-server connection by defining the simulation properties

According to the Figure 19 the user first of all has to define a version, which is the name of the model. After the selection a JavaScript sends a selected variable to the servlet *rainsmap/iframe* which sends a SQL query to the ORACLE database. This query includes the variable in the *where* statement. After the database selection, the servlet gets the selected rows and places them into the combo box on the client side.

The same is repeated twice. After all variables are selected, an OK symbol appears.

The user still has to define the meteorology model, legend, and the emission scenario. These variables are also placed in combo boxes, which are defined when the site is loaded.

After clicking the button “Calculate map query”, a JavaScript looks for all variables in the *iframe* and for the variables as combo boxes.

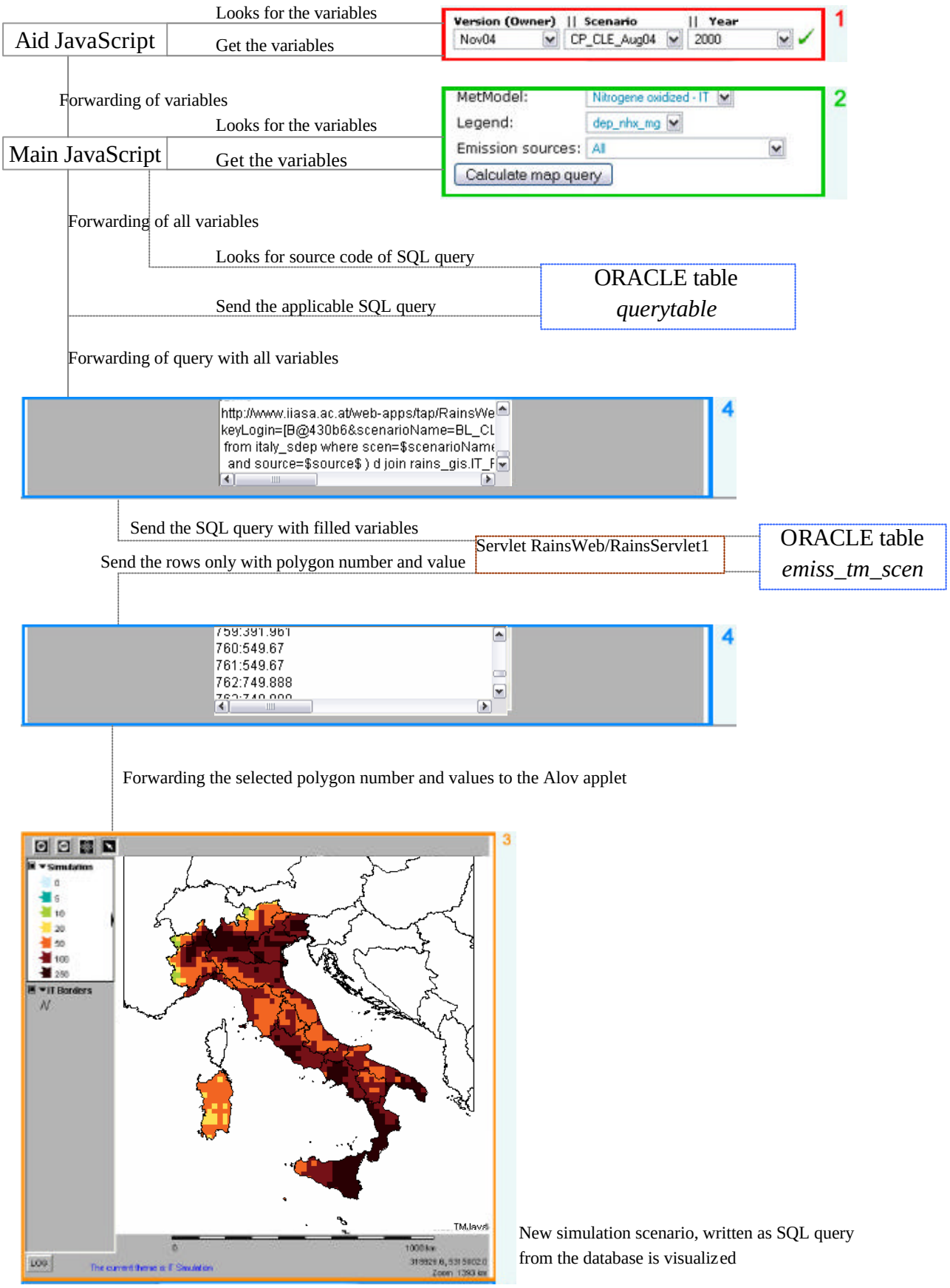

Figure 20: Build-up of a new map according to user demands

After the user has clicked the button "Calculate map query", the *main JavaScript* will call the *aid JavaScript* (compare to Figure 20), which looks for all defined variables in *iframe*. The main JavaScript will get the *iframe* variables back and looks for all variables defined by the user on the main page.

After the main JavaScript has received all variables, it looks in the database for the correct query, which is being adapted for this scenario. The database sends the query back to the JavaScript, which collects all information (query and variables) received and sends them to the IIASA Applet. This Applet connects to the servlet RainsWeb/RainsServlet1 which sends the produced query to the database. The Database answers with all selected rows to the servlet RainsWeb/RainsServlet1 which forwards the information (only the numbers of polygons and the correspondent values) to the IIASA Applet.

The IIASA Applet transforms the data to the correct string, which is sorted by the polygon number and then sends it to the *Alov Applet*.

Alov Applet gets all data and visualizes it on the basis of the legend values and colors.

8.4 New applications added

Creating or loading of a new legend

A registered user to the RAINS online model is allowed to create his/her its own legend for any pollution scenarios. The user interaction interface is shown in Figure 21. The user is allowed to create thematic and topographic legend with any number of levels. The colors should be defined not in RGB but in the CMYK color model as the Figure 19 demonstrates. CMYK colors are prepared for professional printing. To load this legend to the user-selected pollution scenario, the user has only to select the name of the stored legend.

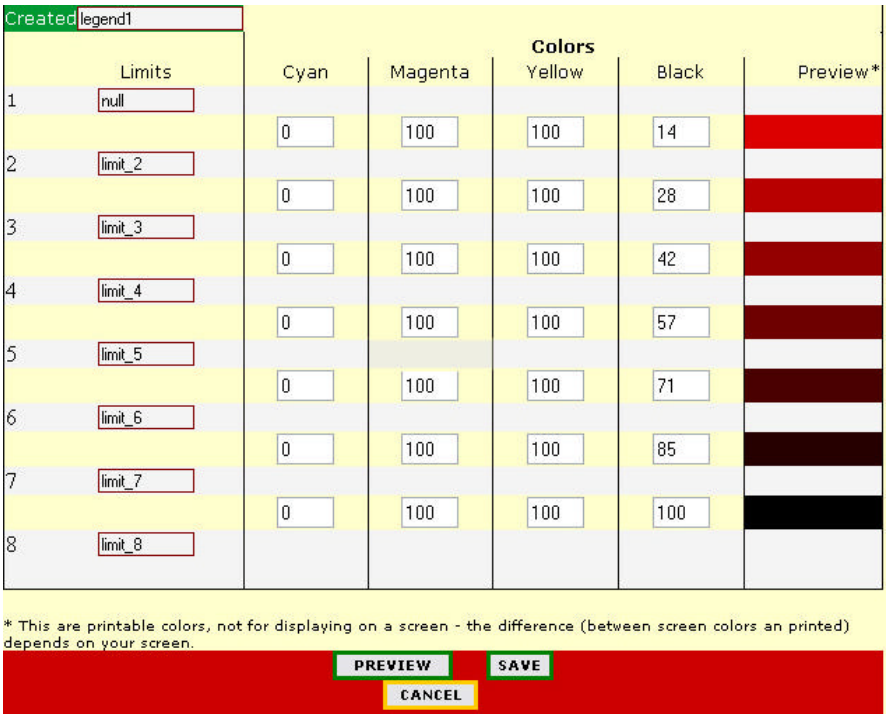

Figure 21: Screen shot of legend creating interface

8.5 Creating a PDF from a displayed map

The user has the possibility to create a PDF for the displayed map and store it further use (i.e., as a presentation, to include in a report or for further editing by other program (i.e., Adobe Illustrator). The sequence and the location of operations is shown in Figure 21.

After a calculation of map sources and often the map has been displayed in the Alov Applet, the client can interact with the map in few ways. These could be zooming or switching the map. If the produced map agrees to the client's wishes it can be transformed from a JAVA Applet window to a PDF file.

According to the Figure 22, to activate it, the client has to push the button "Create map as PDF". First of all, the IIASA Applet sends a query for actual displayed map coordinates, legend values (like colors and values) to the IIASA applet which send back all values. The IMPACT Applet collects all information and sends it to *iText* servlet which, because of the received information, makes a query for needed polygons of the map inside the displayed coordinates. ORACLE answers with all selected rows and send them back to the *iText* servlet, which produces a PDF file.

After the successful generating of the map a PDF, the *iText* servlet sends this new file to the client browser which opens a new window with the PDF file inside. For display of this PDF file, the client needs an *Acrobat Reader*[46] installed.

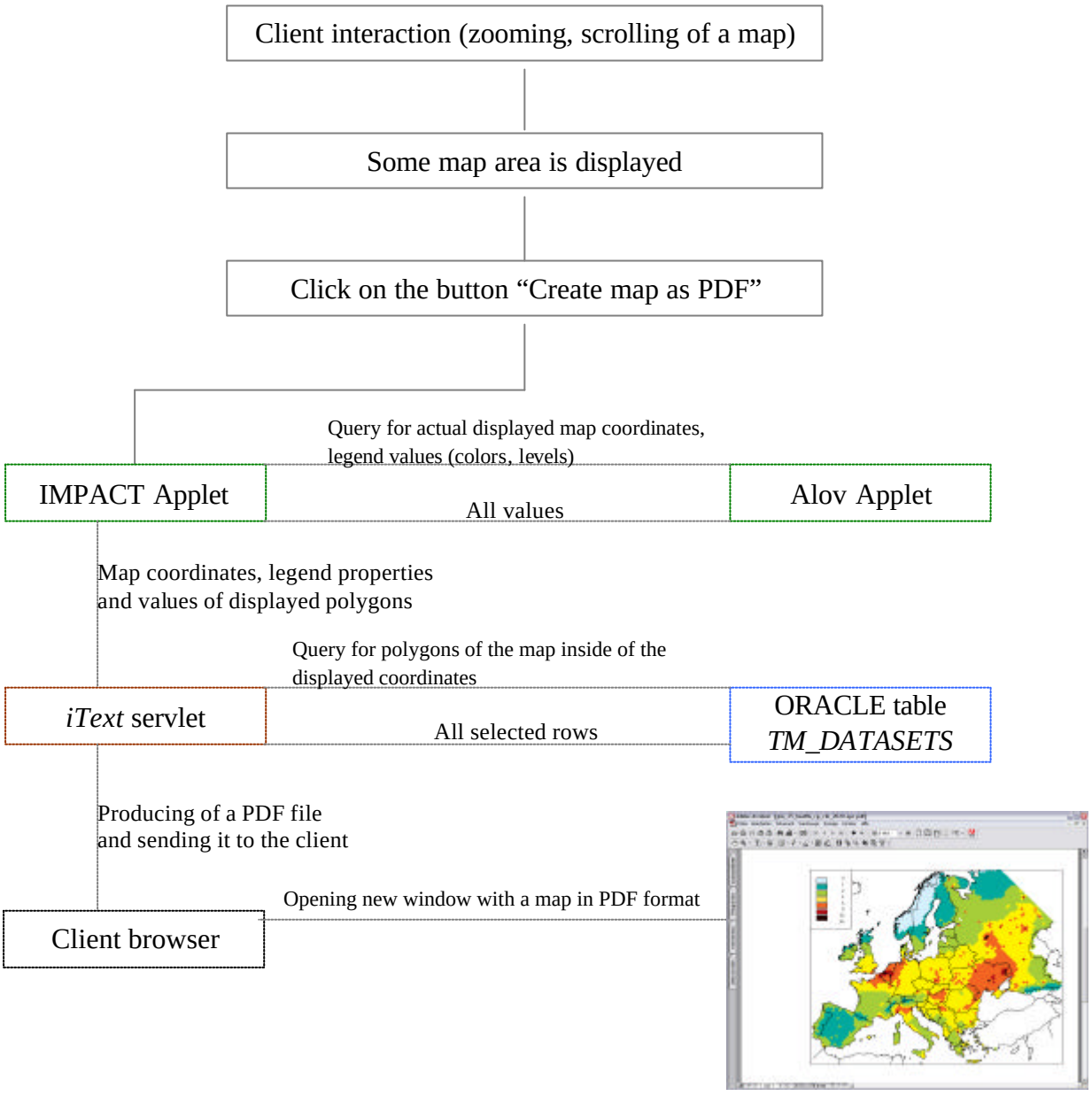

Figure 22: Creation of PDF graphics from displayed maps

Produced maps can be downloaded by the user using only the browser interaction and only in PDF file format.

46 *http://www.adobe.de/products/acrobat/readstep2.html*

8.6 Most important tasks to administrate RAINS IMPACT

For the security reasons, there are some tasks (i.e., adding new datasets) in RAINS IMPACT which are permitted only to the software administrator. This chapter describes some tasks of RAINS IMPACT administration.

Adding a new map

To add a new data set, the source map should consist of two files: a shapefile or MIF file (which are definition formats of the contours of the map) and a table with the data to be displayed (in DBF format). The destination of SQL database connection is already defined, but the user has to insert the user name and password (security check).

Figure 23 shows the administration interface to add a new dataset as map. The necessary inputs are first of all the definitions of the source for the shapefile and DBF file. Afterwards the administrator has to define which database he will update with the new data. The last input required the name of the new map.

Optionally the data can be compressed (this means that the of the polygons will be less precise). These data will be saved in the table TM_DATASETS.

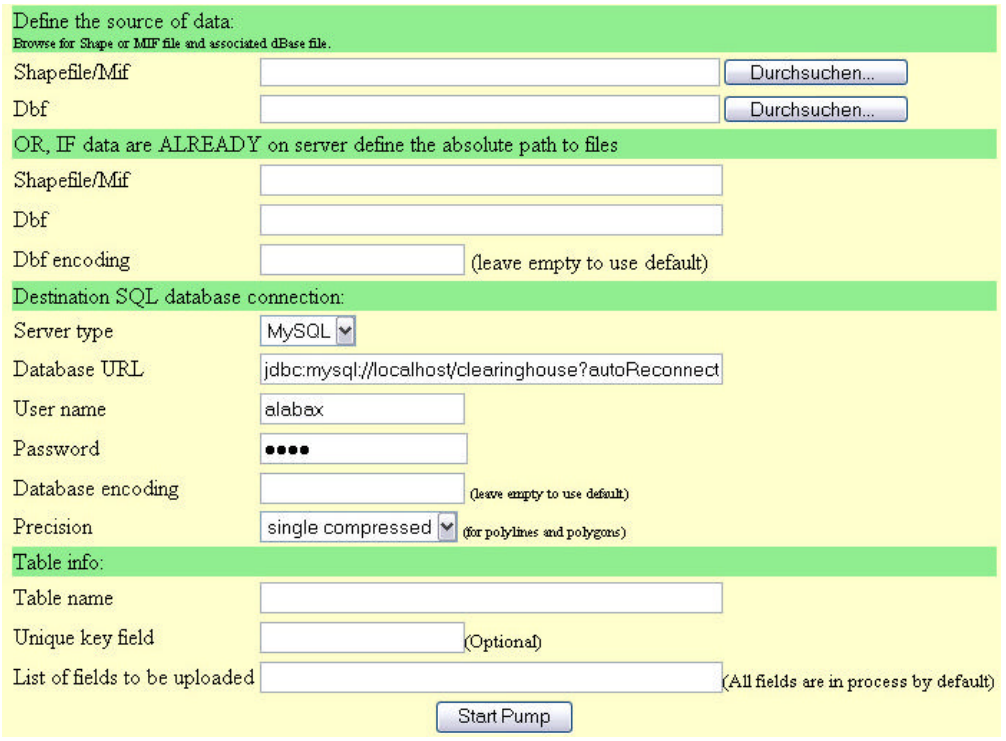

Figure 23: User interface uses to upload a new data set

Creating a new project

A project is a set of more than one data set (map); it includes all information about the legend (colors, levels, symbols), the used maps (datasets), the order of these maps, and the cooperation of all these components.

All this information has to be written as an XML file and uploaded to the ORACLE database, using the user interface. All data will be saved in the table TM_BLOBS (see chapter 6.2.1).

This is an example of source code of a XML project file:

```
01 <?xml version="1.0" ?>
02 <project backcolor="255:255:255" zoomunits="km" usetime="no" zmin="50">
03     <map name="IT Simulation" index="m1"/>
04     <layer label="Simulation" visible="yes" password="yes" >
05     <dataset id="271" full="yes"></dataset>
06     <symbol filled="no" outline="255:0:0"/>
07     <renderer map="m1" type="gradcolor" field="VALUE" >
08       <symbol val="0.00" fill="0:0:255" outlined="no" label="0.00"/>
09       <symbol val="100.00" fill="0:0:255" outlined="no" label="100.00" />
10       <symbol val="200.00" fill="0:219:255" outlined="no" label="200.00"/>
11       <symbol val="400.00" fill="0:255:0" outlined="no" label="400.00"/>
12       <symbol val="700.00" fill="182:255:0" outlined="no" label="700.00"/>
13       <symbol val="1000.00" fill="255:255:0" outlined="no" label="1000.00"/>
14       <symbol val="1500.00" fill="255:0:0" outlined="no" label="1500.00"/>
15       <symbol val="2000.00" fill="255:0:255" outlined="no" label="2000.00"/>
16     </renderer>
17     </layer>
18     <layer label="IT Borders" visible="yes" password="yes" >
19           <dataset id="253" full="yes"></dataset>
20           <symbol filled="no" outline="0:0:0"/>
21     </layer>
22 </project>
```

In the first line is the definition of the XML language standard. The line 02 includes the definition of the project (name, background color, units of the zoom factor and minimal zooming factor). After this follows the definition of the map name. In lines 04-17 this code includes the definitions of the first layer. The important parts of the layer are data-set numbers (identification to find the correct dataset in the database), properties of the legend symbol and the colors, values and the ranges.

8.7 Examples of European maps

Figure 24 represents the deposition of sulphur oxide for the scenario BL_BLE and the year 2000 for Italy.
For more detailed information see *http://www.iiasa.ac.at/web-apps/tap/RainsWeb.*

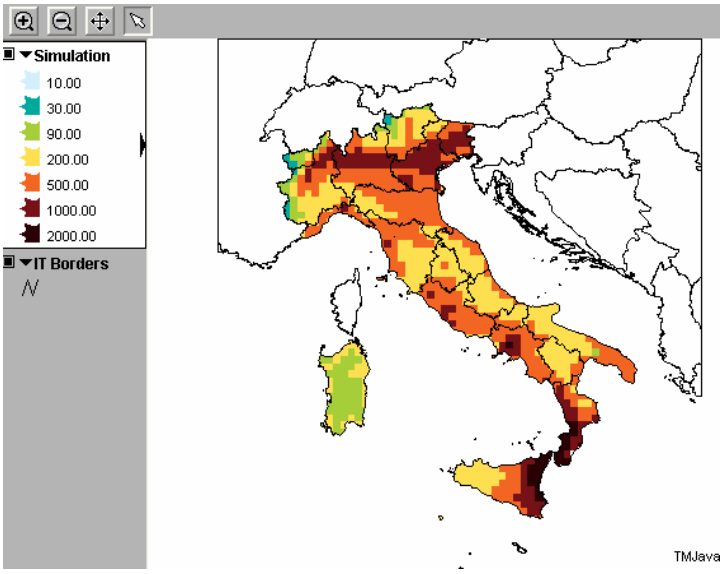

Figure 24: Visualization of a pollution scenario for Italy

Figure 25 represents the deposition of sulphur oxide for the scenario CP_CLE and the year 2000 for the whole Europe region.
For more detailed information see *http://www.iiasa.ac.at/web-apps/tap/RainsWeb.*

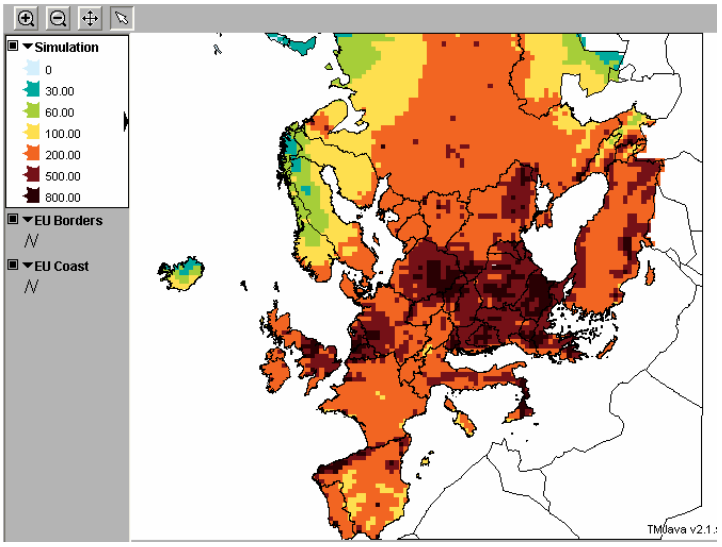

Figure 25: Visualization of a pollution scenario for the whole of Europe

Figure 26 represents the deposition of sulphur oxide for the scenario CP_CLE and the year 2020 for the Netherlands.
For more detailed information see *http://www.iiasa.ac.at/web-apps/tap/RainsWeb*.

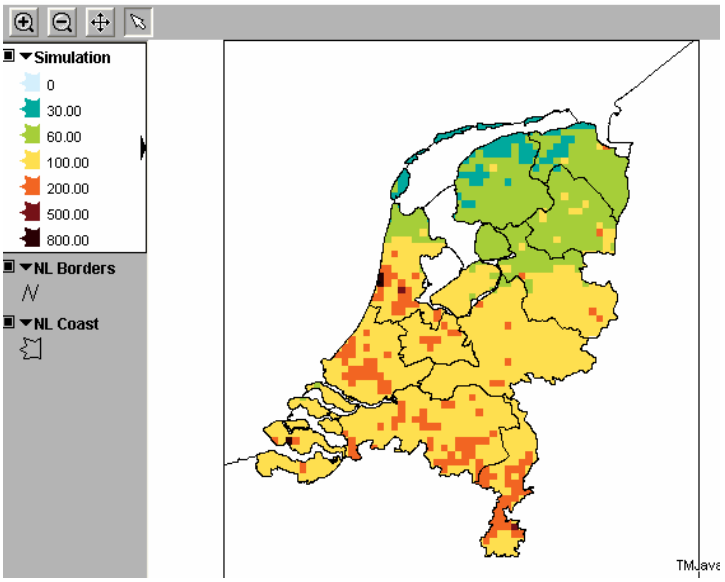

Figure 26: Visualization of a pollution scenario for the Netherlands

9 Summary and conclusions

This study documents the RAINS IMPACT software that has been prepared to visualize the environmental indicators of air pollution mitigation scenarios developed within IIASA's integrated assessment model RAINS. It enables the extraction information about deposition and concentrations of air pollutants for the region under study and combines it with the environmental sensitivities of ecosystems and the impacts of air pollution on human health. Next, it combines it into policy-relevant indicators, and presents them in a graphical form as a series of maps. Within the system several possibilities of defining and extracting the maps have been build, changing legends, scrolling, zooming in and out, and finally printing and exporting them in a standardized output format (e.g., PDF format) for creating the documentation of RAINS model runs or for further use by other software.

The system was developed using a verified and tested open source code. That code was attended and tailored to the specific needs of the model. Because of its open architecture, and the use of well defined current standards, it can be easily modified to include the emerging new Web development standards that are likely to become available within the next two to three years.

As presented in this study results clearly demonstrate that the developed system has met the requirements specified in Section 2. After a final phase of testing, the software will be fully grated into the RAINS Web model and made available to model users. This it is hoped that this study will contribute to a better understanding of air pollution problems in Europe and other regions under study and will suggest efficient ways of combating air pollution in Europe.

Although the software is ready to use and meets the specified requirements, further extensions are needed to increase the model's functionality and transparency. First of all, full model documentation and a user's guide needs to be prepared. Secondly, the capability of serial map production needs to be added in order to quickly document the results of several scenario runs, which are usually necessary to analyze various options of pollution control policies.

10 List of abbreviations

API	– Application Programming Interface
ArcView	– The Geographical Information System
ASP	– Active Server Pages
CMYK	– Cyan + Magenta + Yellow + Black color model
CPU	– Common Process Unit
CSS	– Cascading Style Sheet
DBMS	– Database Management System
DCL	– Data Control Language
DDL	– Data Definition Language
DML	– Data Manipulation Language
DOM	– Document Object Models
EPS	– Encapsulated PostScript
FTP	– File Transfer Protocol
GIF	– Graphics Interchange Format
GIS	– Geographic Information System
GPdf	– GNOME PDF
GPL	– GNU General Public License
GUI	– Graphical User Interface
HTML	– HyperText Markup Language
iframe	– Integrated Frame (part of HTML code)
IIASA	– International Institute for Applied Systems Analysis
J2EE	– JAVA 2 Platform, Enterprise Edition or J2EE
JDBC	– JAVA Database Connection
JPEG	– Joint Photographers Expert Group
JSP	– JAVA Server Page
JVM	– JAVA Virtual Machine
KDE	– K Desktop Environment
LGPL	– Lesser General Public License
Mac OS	– Macintosh Operating System
MySQL	– My Structured Query Language
MrSID	– Multi Resolution Seamless Image Database
PDA	– Personal Digital Assistent
PDF	– Portable Document File
PHP	– Hypertext Pre Processor Language
PL SQL	– Procedural Language Structured Query Language

PNG	– Portable Network Graphic
PS	– PostScript
RAINS	– Regional Air Pollution Information and Simulation model
RGB	– Red + Green + Blue color model
RDBMS	– Relational Database Management System
RSS	– Rich Site Summary, Really Simple Syndication, RDF Site Summary
SQL	– Structured Query Language
SVG	– Scalable Vector Graphics
SWF	– Shock Wave Format
TAP	– Transboundary Air Pollution
URL	– A Uniform Resource Locator
VML	– Vector Markup Language
WAR	– Web ARchive file
W3C	– World Wide Web Consortium
WXS	– W3C XML Schema
XHTML	– Extensible Hypertext Markup Language
XML	– Extensible Markup Language

11 Content of CD

- Computer code developed within the diploma thesis (impact_source.zip)
- *TMJAVA* - original source code (TMJava.tar.gz)
- *TMJAVA* documentation (tm_java_documentation.zip)
- Source code technical documentation (impact_techn_doc.zip)
- Website with the interview on "The future of Flash MX" (Interview The Future of Flash.zip)
- Website with the interview on "The future of the SVG" (O'Reilly Network SVG On the Rise.zip)

12 List of figures

13 List of tables

14 Bibliography

14.1 Books and journal articles

- GULUTZAN, P.: SQL Performance Tuning / Peter Gulutzan and Trudy Peter, Pearcon Education, Inc, 2003

- BENZ B., DURAN J.: XML Programming Bible, Wiley Publishing, Inc, 2003

- TEAGUE J., CAMPBELL M.: SVG for Web Designers, Wiley Publishing, Inc, 2003

- WENZ C.: JAVASript, Galileo Press GmbH, Bonn 2002

- ULLENBOOM C.: JAVA ist auch eine Insel, Galileo Press GmbH, 2003

- NIMA (1997) Geospatial Information Infrastructure Master Plan. Volume 2, Technical Overview. Version 1.0 17 October, 1997.

- Amann, M., Bertok, I., Cofala, J., Gyarfas, F., Heyes, C., Klimont, Z., Schöpp, W. and Winiwarter, W. (2004a) *Baseline Scenarios for the Clean Air for Europe (CAFE) Programme.* International Institute for Applied Systems Analysis (IIASA), Laxenburg, Austria.

- Amann, M., Cofala, J., Heyes, C., Klimont, Z., Mechler, R., Posch, M. and Schoepp, W. (2004b) *The RAINS model. Documentation of the model approach prepared for the RAINS review*. International Institute for Applied Systems Analysis (IIASA), Laxenburg, Austria (*http://www.iiasa.ac.at/rains/review/review-full.pdf*)

- Amann M., Cabala R., Cofala J., Heyes C., Klimont Z., Schöpp W., Tarrason L., Simpson D., Wind P., Jonson J. E. (2004c) The Current Legislation" and the "Maximum Technically Feasible Reduction" cases for the CAFE baseline

emission projection. International Institute for Applied Systems Analysis (IIASA), October 2004.

- Amann M., Bertok I., Cabala R., Cofala J., Gyarfas F., Heyes C., Klimont Z., Schöpp W.Wagner F. (2005) First results from the RAINS Multi-Pollutant, Multi-effects Optimization including Fine Particulate Matter. International Institute for Applied Systems Analysis (IIASA), January 2005

14.2 Internet sites

- *http://www.sun.at*
- *http://oracle.com*
- *http://www.wikipedia.org*
- *http://www.adobe.com*
- *http://www.xml.org*
- *http://www.gnu.org*
- *http://www.carto.net*
- *http://www.geoclip.fr*
- *http://www.npdtechworld.com*
- *http://www.geoinformatik.uni-rostock.de*
- *http://www.lowagie.com/iText*
- *http://www.iiasa.ac.at/Web-apps/tap/RainsWeb*

15 Appendix

RAINS: a model for integrated assessment of air pollution – a short description

The Regional Air Pollution Information and Simulation (RAINS) model provides a tool for analysis of reduction strategies for air pollutants (Amman et al., 1999). The model considers emissions of sulfur dioxide (SO_2), nitrogen oxides (NO_x), ammonia (NH_3), non-methane volatile organic compounds (NMVOC) and particulate matter (PM). RAINS consists of several modules, which contain information on:

- economic activities causing emissions (energy production and consumption, passenger and freight transport, industrial and agricultural production, solvent use etc.)
- emission control options and costs,
- atmospheric dispersion of pollutants and
- sensitivities of ecosystems and humans to air pollution.

It simultaneously addresses health and ecosystems' impacts of particulate pollution, acidification, eutrophication and tropospheric ozone. Thus it creates a consistent framework for multi-pollutant, multi-effect air pollution management. Historic emissions of air pollutants are estimated for each country in Europe based on information collected by international emission inventories (EEA 2001) and on national information (EMEP, 2004). The model also includes projections until 2030. Options and costs for controlling emissions are represented by several emission reduction technologies. Atmospheric dispersion processes over Europe for all pollutants are modeled based on results of the European EMEP model developed at the Norwegian Meteorological Institute (Simpson et al., 2003). The RAINS model incorporates databases on critical loads and critical levels compiled at the Coordination Center for Effects (CCE) at the National Institute for Public Health and Environmental Protection (RIVM) in the Netherlands (e.g., Posch et al., 2004, Hettelingh et al, 2004). For acidification and eutrophication the model estimates the area of ecosystems not protected for each country by comparing deposition on a grid-level against critical loads. For ozone, an estimate is made of the exceedance of critical (damage) thresholds for vegetation (natural ecosystems, agricultural crops) and human health. RAINS contains also a methodology that links air pollution scenarios with changes of statistical life expectancy (Mechler et al. 2002). The present structure of the RAINS model is illustrated in

	SO_2	NO_x	NH_3	VOC	Primary PM emissions
Acidification	√	√	√		
Eutrophication		√	√		
Ground-level ozone		√		√	
Health damage due to fine particles	√	√	√	√	√
	via secondary aerosols				

Table 2: Air quality management as a multi-pollutant, multi-effect problem

The model can be operated in the 'scenario analysis' mode, i.e., following the pathways of the emissions from their sources to their impacts. In this case the model provides estimates of regional costs and environmental benefits of alternative emission control strategies. An 'optimization mode' is also available to identify cost-optimal allocations of emission reductions in order to simultaneously achieve specified deposition and concentration targets.

In this way all four environmental problems from Figure 27 can be addressed simultaneously. The formulation of the optimization problem in RAINS can be found in Amann et al., 2005.

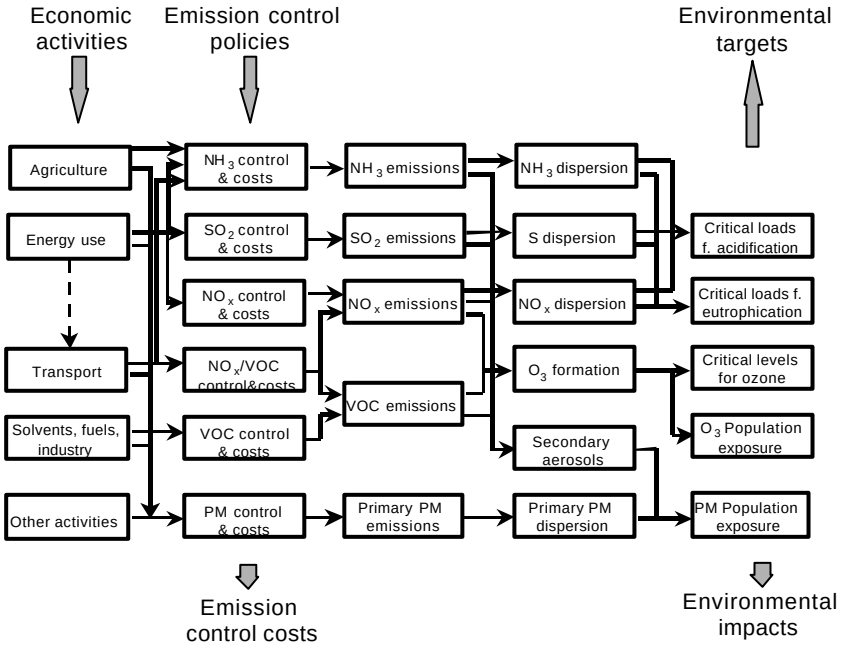

Figure 27: Flowchart of the RAINS model.

The model covers almost all European countries, including the European part of Russia. RAINS incorporates data on energy consumption for 42 regions in Europe, distinguishing about two dozens of categories of fuel use in six major economic sectors. The RAINS database also covers scenarios of non-energy economic activities responsible for air pollution (agricultural production, industrial processes, solvent use etc.). Activity scenarios are an exogenous input to the model.

RAINS calculates emission reductions for control strategies reflecting the current pollution control legislation in Europe. Emission reductions are assumed to be achieved exclusively by technical measures; any feedback of emission controls on economic and energy system is not included. Options and costs for controlling emissions for the various substances are represented in the model by reflecting characteristic technical and economic features of the most important emission control technologies. The model covers several hundreds of technologies. For example, emissions of SO2 can be controlled through the use of fuels with lower sulfur content or through desulfurization of flue gases. Reduction of the emissions from transport can be achieved through implementation of catalytic converters, engine modifications, particulate traps etc. For the reduction of NMVOC a wide range of measures (for instance, recovery of gasoline vapors, use of water-based paints, incineration or recovery of solvents) is available. The model assumes that all control technologies are generally available in every country. However, their actual implementation, and future technical potential and cost takes into account a number of country-specific circumstances like level of technological advancement, installation size distribution, operation regimes, labor costs, etc. Inclusion of those country-specific factors makes it possible to look for optimal cross-country allocation of emission control measures. Information on activities and emission sectors covered by RAINS as well as methodology for emissions and control costs calculation can be found in model technical documentation (*http://www.iiasa.ac.at/rains/databases.html*). Currently a Web-based interface for emissions, control costs and environmental effects calculations is available (compare RAINS Web, 2004). New functions and updates of input data to that interface are under development. More information about the model can be found in Amann et al., 2004b.

References

Amann M., Cabala R., Cofala J., Heyes C., Klimont Z., Schöpp W., Tarrason L., Simpson D., Wind P., Jonson J. E. (2004c) The Current Legislation" and the "Maximum Technically Feasible Reduction" cases for the CAFE baseline emission projection. International Institute for Applied Systems Analysis (IIASA), October 2004.

Amann M., Bertok I., Cabala R., Cofala J., Gyarfas F., Heyes C., Klimont Z., Schöpp W.Wagner F. (2005) First results from the RAINS Multi-Pollutant, Multi-effects Optimization including Fine Particulate Matter. International Institute for Applied Systems Analysis (IIASA), January 2005

EEA (2001) Joint EMEP/CORINAIR Atmospheric Emission Inventory Guidebook. Copenhagen, European Environment Agency.

EMEP (2004) UNECE/EMEP activity and emission database WebDab. Oslo, Norway. (*http://www.Webdab.emep.int*/).

Hettelingh, J.-P. , Posch, M., Slootweg, J. (2004) Critical Loads and Dynamic Modelling Results. CCE Progress Report 2004 No 259101014 Posch, M., Slootweg, J. and, Coordination Centre for Effects, RIVM, Bilthoven, Netherlands

Mechler, R., M. Amann, W. Schoepp and Z. Klimont (2002) Preliminary estimates of changes in statistical life expectancy due to control of particulate matter air pollution in Europe. Oslo, Norway, EMEP: pp. 61 - 89.

Posch, M., Slootweg, J. and Hettelingh, J.-P. (2004) CCE Status Report 2004, Coordination Centre for Effects, RIVM, Bilthoven, Netherlands

RAINS Web (2004) The RAINS Web model. IIASA, Laxenburg, Austria. (*http://www.iiasa.ac.at/Web-apps/tap/RainsWeb*).

Simpson D., Fagerli H., Jonson J. E., 21003: Tsyro S., Wind P. (2003) Unified EMEP Model Description. EMEP Report 1/2003, Oslo, Norway.