TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

D I P L O M A R B E I T

Implementation of the KOMPSAT European Regional Archive Supporting the OGC WebMap-, WebFeature-, and WebCoverageService as well as International Catalog Interoperability

Ausgeführt am Institut für
Analysis and Scientific Computing (E101)
der Technischen Universität Wien

unter der Anleitung von
Prof. Dipl.-Ing. Dr.techn. Dr.rer.nat. Frank Rattay

und an der Abteilung für
Intelligent Infrastructures and Space Applications
der Austrian Research Centers - Seibersdorf research GmbH

unter der Anleitung von
Dipl.-Ing. Dr. Gerhard Triebnig and Dr. Christian Schiller

durch

Stephan Meißl

Weissenbach 90
2371 Hinterbrühl
smeissl@fsmat.at

Seibersdorf im Oktober 2005

Abstract

With computers and Internet being more and more part of our everyday lives, the possibility to provide Earth Observation (EO) data to a greater audience has significantly grown. The development of the KOMPSAT European Regional Archive (KERA) aims to demonstrate and prove an implementation concept for easy access to EO data. The basic results are (i) the offer of enhanced products based on the principles of a high degree of automation, (ii) an easy access to data and services, (iii) and the availability to both expert and non-expert users.

The work of this thesis was facilitated by the well established scientific and technological collaboration between the Korea Aerospace Research Institute[1] (KARI) and the Austrian Research Centers - Seibersdorf research GmbH (Department of Intelligent Infrastructures and Space Applications)[2] (ARC-sr) (Schiller et al., 2004, 2005a,b). The Korean satellite KOMPSAT-1 is used as a test bed for KERA and for examples throughout this thesis.

Considering the archive it is necessary to store all required data and metadata in an appropriate way. Hereby, an Object-Relational Database Management System (ORDBMS) is used for KERA. The design, considering existing standards and specifications, and the implementation of this database (ARCS SATdata Catalog) are presented in chapter 2. Additionally software used, problems occurring, processes triggered to fill the database, as well as the installation procedures are described. An additional section deals with speed issues relevant for the ARCS SATdata Catalog.

Additional developments for KERA are achieved by applying the Open Geospatial Consortium, Inc. (OGC) Web Services (OWS) WebMapService (WMS), WebFeatureService (WFS), and WebCoverageService (WCS). These standardized and specified online web services are used to access the data in various ways over the Internet. They are explained by illustrative examples in chapter 3.

Furthermore, different access points to the EO data stored in KERA are explained in chapters 4 and 5. First, the method implemented and located at

[1] URL: http://www.kari.re.kr
[2] URL: http://www.space-applications.at

ARC-sr is introduced followed by access via the Service Support Environment (SSE)-Portal and eoPortal, both hosted by the European Space Agency (ESA).

Finally, chapter 6 gives a presentation of the results of this thesis. It summarizes the work and provides a scope for further research developments. Making satellite data more accessible and reducing the necessity of investment in software and time to learn the handling, will improve the acceptance and increase the number of users. These improvements might additionally enlarge the various possibilities to use satellite data in miscellaneous applications.

Kurzfassung

Seit Computer und das Internet immer mehr Teil des alltäglichen Lebens sind, ist es viel einfacher geworden Erdbeobachtungsdaten einem breiteren Publikum zugänglich zu machen. Das Ziel der Entwicklung des regionalen europäischen KOMPSAT Archivs (KERA) (Englisch: KOMPSAT European Regional Archive) ist die Demonstration und die Erprobung des Konzepts zur Umsetzung eines Erdbeobachtungsdatenkatalogs und des einfachen Zugangs zu diesem. Die wesentlichen Ergebnisse sind (i) ein aufgrund des hohen Grads der Automatisierung erweitertes Angebot an verbesserten Produkten, (ii) ein einfacher Zugang zu Daten und Diensten (iii) und die Erreichbarkeit sowohl für erfahrene als auch für unerfahrene Benutzer.

Die Enwicklung des Archivs ist aufgrund der gut eingeführten wissenschaftlichen und technologischen Zusammenarbeit zwischen dem koreanischen Luft- und Raumfahrts Forschungsinstitut[3] (KARI) (Englisch: Korea Aerospace Research Institute) und der Abteilung für Intelligente Infrastrukturen und Weltraumanwendungen der Austrian Research Centers - Seibersdorf research GmbH [4] (ARC-sr) möglich (Schiller et al., 2004, 2005a,b). Der koreanische Satellit KOMPSAT-1 ist die Grundlage dieser Arbeit und wird mehrfach für Beispiele herangezogen.

Das Archiv muß alle benötigten Daten und Metadaten auf eine adäquate Art abspeichern. Für KERA wird ein objektrelationales Datenbankmanagementsystem (ORDBMS) verwendet. Das Design, das existierende Standards berücksichtigt, und die Implementierung dieser Datenbank, ARCS SATdata Catalog, wird im Kapitel 2 beschrieben. Dieses Kapitel beinhaltet auch Beschreibungen der verwendeten Software, der auftretenden Probleme, der zur Füllung der Datenbank notwendigen Prozesse und der Installation. Ein Abschnitt ist auch der Steigerung der Geschwindigkeit der Datenbank gewidmet.

Die darauf aufbauenden Entwicklungen für KERA sind die Web Dienste WebMapService (WMS), WebFeatureService (WFS) und WebCoverageService (WCS) des Open Geospatial Consortium, Inc. (OGC). Diese online Web Dienste sind von OGC standardisiert und klar spezifiziert. Sie werden angewendet um verschiedene Möglichkeiten des Zugangs zu den Daten über das Internet zu

[3] URL: http://www.kari.re.kr
[4] URL: http://www.space-applications.at

ermöglichen. Genaue Beschreibungen und viele Beispiele dazu finden sich im Kapitel 3.

Die verschiedenen Zugangsmöglichkeiten zu den Erdbeobachtungsdaten von KERA werden in den Kapiteln 4 und 5 beschrieben. Zuerst wird ein Zugang zu dem ARC-sr eigenen Server vorgestellt. Danach werden die beiden Zugänge über das SSE-Portal (Englisch: Service Support Environment) und über das eo-Portal vorgestellt. Beide Möglichkeiten werden von der europäischen Weltraum Behörde (ESA) (Englisch: European Space Agency) betrieben.

Abgerundet wird die Arbeit durch die Zusammenfassung und den Ausblick auf mögliche Weiterentwicklungen (Kapitel 6). Durch KERA werden Satellitendaten leichter erreichbar und dadurch Zeit und Kosten für deren Nutzung gesenkt. Diese Verbesserungen helfen möglicherweise die Akzeptanz und die Anzahl der Benutzer von Satellitendaten deutlich anzuheben. Sie steigern hoffentlich auch die Verwendungsmöglickeiten von Satellitendaten in verschiedensten Anwendungen.

Contents

Acknowledgment

Starting in summer 2002 as an intern of the department of Intelligent Infrastructures and Space Applications of Austrian Research Centers - Seibersdorf research GmbH (Department of Intelligent Infrastructures and Space Applications) (ARC-sr) my interest in Earth observation research has been roused and grew until the summer 2004 the first ideas regarding my thesis appeared. These ideas became clearer and more distinct during fall. After writing a thesis proposal and finding a professor supervising it, I started with the thesis work in January 2005 at ARC-sr.

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. Especially, I want to thank DI DDr. Frank Rattay for supervising and for the possibility to write the thesis at the Institute for Analysis and Scientific Computing (E101) of the Vienna University of Technology, DI Dr. Gerhard Triebnig and Dr. Christian Schiller for supervising and for the chance to work and write my thesis in such an interesting field of research, and all my colleagues at ARC-sr for helping me through some difficulties during the writing.

I also want to thank all those who read the thesis and provided me with helpful comments, especially to DI Sonja K. Ulreich and DI Andrea Kolassa, MSc. Finally I want to thank my whole family for their loving support during my whole study.

I hope that this thesis can help to increase the acceptance and number of users and use cases of satellite data.

Chapter 1

Introduction

Where? This question is fundamental to humanity. In many different contexts it is important to ask this question. For example: "Where shall the new highway be build?", "Where should a new store, a new petrol station, a new park, etc. be located?" Hence it is not surprising that a huge percentage of available information and data has a spatial (geographical) component.

Maps are an appropriate way to view this information since they are human readable. It is almost impossible for humans to interpret a list of coordinate values as fast and accurate as a map. Usually, we think of paper maps. Since today's computers are getting faster and smaller, those maps are changing more and more from static paper maps to highly dynamic and fast adaptable digitally stored computer maps. Another advantage of these computer maps is the easy way to copy and distribute them. Hereby, the Internet gives the adequate platform for making them available to the whole world in reasonable time.

The data processed at the Austrian Research Centers - Seibersdorf research GmbH (Department of Intelligent Infrastructures and Space Applications)[1] (ARC-sr) (see glossary at page 85 for explanations on acronyms and technical terms) are images derived from earth observation satellites. Currently nearly all data arrives from the first Korean Multi Purpose Satellite (KOMPSAT) called KOMPSAT-1. The archive will be expanded by data from the future KOMPSAT-2 satellite. Both missions are developed and operated by the Korea Aerospace Research Institute[2] (KARI). This is possible due to a well established scientific and technological collaboration between KARI and ARC-sr (Schiller et al., 2004, 2005a,b).

The KOMPSAT-1 satellite collects high-resolution (HR) imagery at 6.6 m spatial resolution. The sensor, called Earth Observing Camera (EOC), delivers panchromatic images with a swath width of approximately 17 km. The future KOMPSAT-2 platform with its very-high-resolution (VHR) Multi-Spec-

[1] URL: http://www.space-applications.at
[2] URL: http://www.kari.re.kr

tral-Camera (MSC) will provide imagery with one panchromatic channel with a spatial resolution of 1 m and four multi-spectral channels with 4 m spatial resolution. The swath width will be approximately 15 km. Figure 1.1 shows the KOMPSAT-1 satellite. Figures 1.2 - 1.4 give examples of KOMPSAT-1 images and figure 1.5 shows a simulated KOMPSAT-2 image using the IKONOS satellite.

Figure 1.1: KOMPSAT-1 satellite, top: in flight sketch, bottom: on shaking-test platform

At ARC-sr the images arrive as so-called "raw-data" images. They are called "raw" because no additional processing, like orthorectification or classification, has yet been added to them. Actually those images are not really "raw-images"

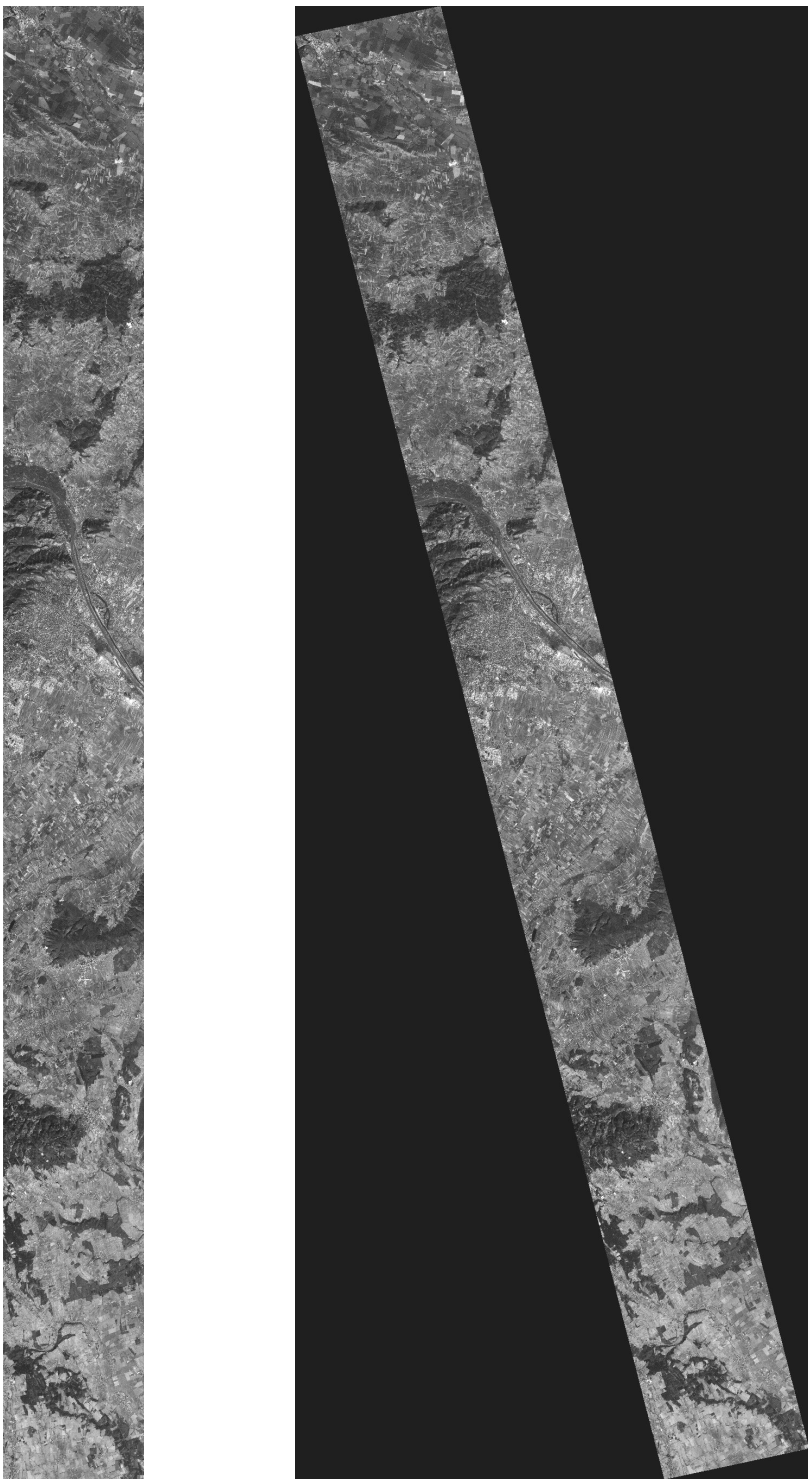
because sensor and platform correction and some scattering correction have already been applied. These corrections are done at KARI since nobody else has the information required. However this is the only processing performed on the tiff-files and that is the reason why they are called "raw". Figures 1.2(a) and 1.3 show examples in hereby. Figure 1.2(a) pictures the whole image as it reaches ARC-sr and figure 1.3 a part of that image, showing the center of Vienna, Austria. The first problem occurring is the storage and archiving of this data. One raw-image is likely to need about 70 $MB(Megabyte)$ of storage place on hard disc but can reach up to 300 MB as well. 70 MB of storage place are approximately equal to 180 km of recorded Earth surface. It is also important to remember which images are stored where in order to be able to search and access them easily.

Today the use of digital geographic information is expanding because people with various backgrounds outside geographic science and information technology are capable of producing, enhancing and modifying it. However not everybody who wants to use these images has the knowledge of processing and naming them. Therefore, additional data about the images, so called metadata, needs to be stored and made available. In order to store all these information, a Database Management System (DBMS) is required. Such systems not only collect all data centralized and allow sophisticated searches but also improve the speed of searches. Most of the searches will be, or at least include, a spatial dimension. Hence, it is also important to improve and increase the possibilities of the database by using spatial extensions.

After first inputs to the database have been made the following step is to "orthorectify" the images. This implies to add geospatial information to them in order to get knowledge where they are actually located on the earth's surface and to make a spatial search possible. This is done by assigning geographic coordinates to image-pixels, i.e. exploring Ground Control Points (GCP) or Tie Points of known locations. This is very time consuming, especially since there is no continuous information resource available to provide such information covering Europe. Images aren't only shifted or/and rotated but also are some image distortions corrected. Figures 1.2(b) and 1.4 show the same images as figures 1.2(a) and 1.3 but are orthorectified using the Universal Transverse Mercator (UTM) projection.

The images are now stored in the file system as Geographically Registered Tagged Image File Format (GeoTIFF) formated files. The size of the former files with 70 MB increases to about 350 MB due to the black areas at the corners shown by figure 1.2(b). Another significant benefit of the orthorectification is the chance of combining different data-sets in one map. Those data-sets can be stored in the same storage place on one computer or on any connected computer. They can even be distributed over the Internet, which leads directly to the next topic.

It is of significant interest of the geospatial community to enable interoperability among and between diverse geospatial data stores, services, and applica-



(a) raw (b) orthorectified

Figure 1.2: KOMPSAT-1 “raw” and derived “orthorectified” (using UTM) image showing the city of Vienna, Austria in the middle

Figure 1.3: Part of the image of figure 1.2(a) (raw image) showing the city-center of Vienna, Austria

Figure 1.4: Part of the image of figure 1.2(b) (orthorectified image, using UTM) showing the city-center of Vienna, Austria

Figure 1.5: Simulated KOMPSAT-2 image using the IKONOS satellite showing St. Stephens Cathedral in the city-center of Vienna, Austria (©: Space Imaging)

tions. The Open Geospatial Consortium, Inc.[3] (OGC) is an organization trying to design interface and encoding specifications to enable this goal. There are some problems to be solved. For instance a lot of geospatial data is available via the Internet but most of these data are stored in different data formats using different data models, coordinate reference systems, geometry models, etc. Thus, it is rather difficult to share spatial data easily. The Frequently Asked Questions (FAQ) section of OGC's homepage (OGC, FAQ) says:

> "OGC manages a consensus process in which specifications for common software interfaces and encodings are developed to enable users to maximize the value of past and future investments in geoprocessing systems and data. That general need points to three more specific classes of user needs:
>
> 1. The need to share and reuse data in order to decrease costs (avoid redundant data collection), get more or better information, and increase the value of data holdings.
> 2. The need to choose the best tool for the job, and to reduce technology and procurement risk (i.e., the need to avoid being locked in to one vendor).
> 3. The need for more people with less training to benefit from using geospatial data in more applications: That is, the need to leverage investments in software and data."

These needs are also recognized at the Austrian Research Centers and therefore some OpenGIS® Specifications are implemented (Schiller et al., 2005a). A Geographic Information System (GIS) is mostly defined as software system for managing spatial data and associated attributes. It is capable of integrating, storing, editing, analyzing, and displaying this data. The implemented specifications include the WebMapService (WMS), the WebFeatureService (WFS), and the WebCoverageService (WCS). These services give access to the KOMPSAT European Regional Archive (KERA) in various ways. While the WMS distributes pictures, the WFS can be used to give access to the actual vector-data e.g. thematic products, whereas the WCS presents a way to provide access to the whole raster-data. The software used is the OpenSource MapServer[4], originally developed at the University of Minnesota, which provides the required services.

The MapServer system also includes PHP/MapScript that allows the popular scripting language "PHP: Hypertext Preprocessor" (PHP) to access the MapServer C Application Programming Interface (API). There are similar possibilities for Perl, Phyton and Java, but because MapServer is mostly used for Internet applications PHP is the best choice to use. PHP also provides an easy access to the database management system (used e.g. for a gazetteer) and

[3] URL: http://www.opengeospatial.org
[4] URL: http://mapserver.gis.umn.edu

PHP/MapScript adds a lot of spatial functions. With the PHP/MapScript, PHP and some JavaScript functions an Internet client was implemented at ARC-sr to show the possibilities of the services. It can be explored with every standard web browser e.g. like Mozilla's Firefox at http://spacey.arcs.ac.at. The basic functionality includes zoom in and out, select multiple layers and display them together, query some vector layers for additional data, etc. The client is also able to integrate remotely provided geospatial data from other WMS via the Internet with the cascading function. For ARC-sr it is important to implement the WMS fully on the server side making it unnecessary to install any software (e.g. plug-ins, applets, etc.) at the client side.

A short response time is very important for the WMS, WFS and WCS. Thus the used database, which stores the metadata, some vector-based datasets, and also data about the large raster images, plays a decisive role. To increase speed and performance, the large raster images are tiled into smaller pieces and shapes are created for their outline. These shapes are saved on the database. After the database has been searched for the required area (based on the outline vector shapes) only the actually required raster images have to be loaded and processed. The tilling has the additional advantage of saving only tiles which are not black. There are also so-called "overviews" which are images at half resolution. The advantage of these can be seen at figures 1.2(b) and 1.4. The information seen in the second figure is also stored in the first one but invisible because of the printer's or monitor's resolution. Together with all these speed improvements the storage place required for one image of originally 350 MB is reduced to approximately 250 MB. This is still more than the initial 70 MB but we have gained orthorectification and performance in speed.

It is intended to show an example of an OGC compliant Earth Observation (EO) application with the KOMPSAT European Regional Archive (KERA) at ARC-sr. This implies that it is not only accessible for search requests via the local client and services but also via different access points i.e. via the eoPortal[5] and SSE-Portal[6] (Service Support Environment (SSE)) provided by the European Space Agency (ESA). Therefore, it was important to build the database following the international standards, specifications and rules provided by ESA.

Obviously dealing with interoperable geospatial data, like satellite derived images, involves a lot of requirements. The need for a spatial enabled database, the need for standardized distribution mechanisms like WMS, WFS and WCS, the need for the possibility to search and access the data from different portals, the need for interoperability among and between data stores, services, and applications, etc. This thesis wants to give an overview how these needs can be satisfied considering the methods and strategies applied at the Austrian Research Centers.

The second chapter introduces and describes all database related issues, like

[5]URL: http://catalogues.eoportal.org
[6]URL: http://services.eoportal.org

geospatial metadata, database design at ARC-sr and standards conformance. Subsequently chapter 3 deals with the OGC services WMS, WFS, and WCS. After the definition of the services the software MapServer will be explained. The third chapter covers the implementation of a client to demonstrate and extend the possibilities of the services using PHP/MapScript, PHP and JavaScript. After a discussion about service integration, i.e. into the SSE-Portal and the eoPortal, the final chapter will offer conclusions, the "lessons learned", and some future ideas.

In the back, there is a glossary included (page 85) which might appear helpful for the reader. Acronyms and technical terms are summarized there.

Chapter 2

Metadata Catalog Service

As already mentioned in the introduction (page 3), it is very important for every data warehouse to store metadata, mostly defined as "data about data" (Longley et al., 2001; ISO/TC 211, 2003). The biggest data collection is good-for-nothing if nobody knows which and how data is stored. The best example to demonstrate the need for metadata is a library. If somebody wants to find a specific book, knowing its title, he needs to find out where it is situated. This is a rather easy case since the title of the book is known however the need for metadata already exists. What happens when the title is not known a priori, i.e. books concerning a specific topic, books referencing a specific one, etc.? There is always a need for metadata involved to find a specific book. When dealing with satellite derived images stored digitally one faces similar problems.

In former days there existed one file card for every book, storing all interesting information regarding this specific book. Nowadays, computers are used to store this metadata. They have several advantages over file cards. Computers are a lot quicker than humans, they need fewer storage place for the same amount of data, they offer a huge variety of sophisticated searches and due to the Internet they can be accessed and searched from nearly every place in the world.

The Internet and it's availability offers the possibility to search more than one storage place at the same time. However there are still some difficulties arising with such searches. If two or more servers are queried at the same time they need to "talk the same language". So there is a big need for standardization on metadata storage and access. The following section introduces and explains the difficulties with geospatial metadata and gives an overview about some standards regarding this metadata. After this introduction a section regarding Database Management Systems and another dealing with the difficulties of geospatial data follows. The chapter ends with a detailed description of the Metadata Catalog Service implemented at ARC-sr called "ARCS SATdata Catalog".

2.1 Geospatial Metadata

"Spatial is special!" written by Longley et al. (2001) clearly points out the uniqueness of spatial data. But what means spatial and geospatial, and what are the differences? Spatial refers to any space, not only the Earth's surface like geospatial or geographic does. The spatial component of some information is the description of "Where", i.e. where did something happen, where is something located, etc. If this refers to the Earth's surface the term geospatial is used to describe the information.

In former days geospatial data was mostly viewed as static paper maps. Nowadays most of this data is digitally stored in computers and a lot of new possibilities for its usage arise. There are a lot of different approaches how to get the reality into a digital representation and how to store it. Two basic concepts are vector and raster representation. They derive from object and field view of reality. Generally, the object view results in a vector representation and field view in rasters. In object view reality is identified by objects like houses, streets, etc. whereas field view treats reality like a continuous two dimensional space. Using vector representation objects are stored as geometrical primitives such as points, lines, polygons, and sometimes curves. Raster representation underlies a grid of pixels. For every pixel the value of an attribute like elevation or land use class (e.g. as color code) is saved.

Analyses beyond that point are not within the scope of this thesis but are discussed in various other articles, e.g. (Longley et al., 2001, chap. 3).

Geographic data is generally an attempt to describe the real world however every description is always an abstraction and always just one view. Without a description of the data only the producer will be familiar with it. To all the others the data is almost worthless. To ensure that data is not misused or misinterpreted all assumptions and limitations to the data have to be documented in the metadata. Therefore, the metadata should provide information about identification, extent, quality, spatial and temporal schema, spatial reference, and distribution relevant information about the data.

ISO/TC 211 Geographic information/Geomatics[1] (ISO/TC 211) is the ISO Technical Committee (ISO/TC) responsible for defining international standards in the area of digital geographic information. It is part of the International Organization for Standardization (ISO). The committee's work includes the international standard (ISO/TC 211, 2003) on metadata and it cooperates closely with the Open Geospatial Consortium, Inc. (OGC). Thus, not surprisingly, their abstract specification (OGC AS, Topic 11, 2000) is the same as the ISO 19115 standard. OGC defines itself as "a non-profit, international, voluntary consensus standards organization that is leading the development of standards for geospatial and location based services." (See 3.1 for more information on OGC.) At ARC-sr, it is also important to take ESA's (SSE ICD 1.4, 2005)

[1] URL: http://www.isotc211.org

into account to make the KOMPSAT archive searchable and accessible via the SSE-Portal and the eoPortal.

As stated above the metadata will include information about:

- identification: Basic information like title, contact, constraints, etc.
- extent: Covered area.
- quality: Quality information like source, processing, used algorithms, etc.
- spatial and temporal schema: Schemas used for spatial and temporal information.
- spatial reference: Spatial reference system used (projection, ellipsoid, datum, etc.).
- distribution: Information for obtaining data (distributor, data-format, medium, etc.)

2.2 Database Management System (DBMS)

All this digital geographic data needs to be stored somehow on the computer. The best way to do to this is to use a database and a Database Management System (DBMS). Nowadays, such systems are commonly used and have many advantages. A significant benefit is the physical data independence i.e. the ability to modify the physical schema without changing the logical schema. The physical or data schema is a fully attributed definition. Detailed attributes are defined for each entity used to create a database. The logical or object schema is a data model of a specific problem domain, which does not include the design considerations and physical storage parameters found in a physical schema. A logical schema contains entities made up of attributes connected by relations. Examples of both schemas can bee seen in sections 2.4.1.1 and 2.4.1.2.

An entity is a representation of a discrete object. Concerning relational databases, "entity" is often similarly used as "table". Attributes are inherent properties in an entity or associated with that entity. For example to store an archive for a library the database needs the entities "book", "thesis", etc. The entity "book" needs attributes like "author", "title", "year", etc.

Other improvements of DBMS are to control the concurrent access by multiple users, to prevent data inconsistency, to provide the possibility of integrity checks, to prevent redundancies, to make local and remote access easier, to provide recovery strategies in case of server crash, and to control different access privileges. More information can be found in e.g. (Kemper and Eickler, 2001).

At ARC-sr it was decided to use PostgreSQL[2] as database management system because of various reasons. First, PostgreSQL is an OpenSource Object-Relational Database Management System (ORDBMS). It is based on POSTGRES, Version 4.2, developed at the University of California at Berkeley Computer Science Department and it supports a large part of the new SQL:2003 standard (see next clause). Second, PostgreSQL can be extended with software like PostGIS[3], which adds support for geographic objects to PostgreSQL. In effect, PostGIS "spatially enables" the PostgreSQL server. PostGIS follows the "OpenGIS® Simple Features Specification for SQL" (OGC, SFS for SQL, 1999). The versions used for this project at ARC-sr are PostgreSQL 8.0.1 and PostGIS 1.0.0.

The Structured Query Language (SQL) is used to create, modify and retrieve data from relational database management systems. This American National Standards Institute (ANSI) and ISO standard was first published in 1986 and subsequently revised several times. The SQL:1999 standard added object-oriented features, the current version is SQL:2003. The SQL structure allows users and programmers to be less familiar with technical details, e.g. data storage, and relatively more familiar with the information contained in the data. The basic operations are data retrieval, data manipulation, data transaction, data definition, and data control:

- retrieval: Get data out of the database. SQL-command: `SELECT`, mostly used like: `SELECT . FROM . WHERE ..`
- manipulation: Change stored data. SQL-commands: `INSERT`, `UPDATE`, and `DELETE`.
- transaction: Grant data integrity. SQL-commands: `BEGIN`, `COMMIT`, and `ROLLBACK`.
- definition: Define data structures like tables, views, etc. SQL-commands: `CREATE` and `DROP`.
- control: Manage access privileges, like read (`SELECT`) or modify (`INSERT`, `UPDATE`, and `DELETE`), for different users and groups of users. SQL-commands: `GRANT` and `REVOKE`.

Another advantage of a DBMS in conjunction with SQL is the simplification of the development of scripts used for searches e.g. via the SSE-Portal and the eoPortal. Such scripts are easily able to connect to the database and use the SQL command `select` to retrieve the data required.

[2]URL: http://www.postgresql.org
[3]URL: http://postgis.refractions.net

2.3 Geospatial Search

The most common search, dealing with satellite derived images, is the simple spatial search (e.g. Which images are showing Vienna?, Which images are within Europe?). What has to be done to get the correct answers to these questions? First the outline of every image has to be stored as object in the database. The PostGIS object POLYGON is the appropriate data-type. The four corner coordinates of the image are stored in simple longitude/latitude coordinates using the World Geodetic System (WGS84) (defined in the next section).

2.3.1 Definitions

Longitude describes the location of a place on earth east or west of a north-south line called the prime meridian. Longitude is defined as the angle between this prime meridian and the location, ranging from 0° at the prime meridian to +180° eastward and −180° westward, or sometimes as ranging from 0° − 360° going eastward. Since there is no natural starting position for longitude a reference meridian had to be chosen. This was done at Greenwich, London, UK. Vienna for example has a longitude of ∼ 16.37°.

Dealing with latitude is a bit different and more sophisticated. Latitude describes the location of a place on earth north or south of the equator. The equator is an imaginary line drawn around the earth halfway between the poles. Latitude is now defined as the angle between the equator and the location, ranging from 0° at the equator to +90° northward and −90° southward. Since earth is slightly flattened by its rotation, geographic latitude is measured assuming earth as a spheroid rather than a sphere (see figure 2.1).

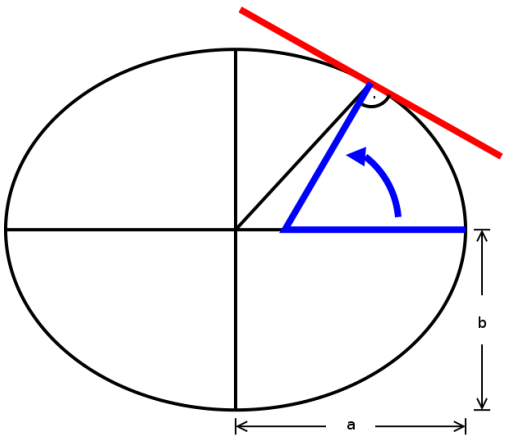


Figure 2.1: Definition of latitude, as the angle (shown in blue) between equator and the line drawn perpendicular to the ellipsoid (red).

The spheroid is obtained by rotating an ellipse. The used reference ellipse is

WGS84 which is defined with an equatorial semi-major axis of $a = 6378137\ m$ and a flattening $f = 1/298.257223563$. The flattening f is defined as: $f = \frac{a-b}{a}$ where b denotes the polar semi-minor axis. For Vienna this means a latitude of $\sim 48.2°$.

According to the definition of a coordinate system (CS) the only thing left for the full definition of a coordinate reference system (CRS) is the datum. The datum is a parameter or set of parameters that define the position of the origin, the scale, and the orientation of a coordinate reference system. The used datum is named WGS84 like the ellipsoid.

The projection based on this coordinate reference system has got the unique *EPSG* number 4326 which is assigned by the European Petroleum Survey Group[4] (EPSG) and widely used in the geographic community (e.g. by OGC). PostGIS uses the cartographic projections library PROJ.4[5] for projection transformations. They both use the EPSG numbers. PROJ.4 is distributed under the MIT License which allows reuse for open source or proprietary software. The PROJ.4 definition of *EPSG* 4326 reads as:

```
<4326> +proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs  <>
```

2.3.2 PostGIS

PostGIS has its own table storing all projection information which is called `spatial_ref_sys` (according to (OGC, SFS for SQL, 1999)). The information stored in this table for *EPSG* 4326 is the same as above but in a more detailed form:

```
GEOGCS["WGS 84",DATUM["WGS_1984",SPHEROID["WGS 84",6378137,298.25
7223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIME
M["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.0174532
9251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]]
```

which additionally describes the used spheroid and units of measurement. PostGIS uses its own unique numbering system. These numbers are called Spatial Reference System Identifier (SRID). Thankfully, the same numbers as for EPSG can be used.

Returning to satellite images and their outline it is important to know how to store them in the database. PostGIS uses the Well-Known Text (WKT) and the Well-Known Binary (WKB) forms (see (OGC, SFS for SQL, 1999)) to represent geographic objects. The WKT representation of the geometry from the outline-polygon of the KOMPSAT-1 image (Figure 1.2) is:

[4]URL: http://www.epsg.org
[5]URL: http://www.remotesensing.org/proj/index.html

```
POLYGON((16.02063625 48.86594772,16.26286888 48.89667965,16.81641
535 47.30641733,16.58351857 47.27637656,16.02063625 48.86594772))
```

PostGIS provides the functions `GeometryFromText(text WKT, SRID);` and `asText(geometry);` to store and retrieve data in a human readable form. The following SQL-command stores the spatial information of the already known image with the name "eoc06182_20010216T091614" (naming convention see section 2.4.1):

```
INSERT INTO <table_name> VALUES ('eoc06182_20010216T091614',Geome
tryFromText('POLYGON((16.02063625 48.86594772,16.26286888 48.8966
7965,16.81641535 47.30641733,16.58351857 47.27637656,16.02063625
48.86594772))',4326));
```

Once the data for all images is stored in the database it is easy to answer the questions posed at the beginning of this section (page 15). What has to be done for the question: "Which images are showing Vienna?"? The coordinates of Vienna are already known (Lon: $\sim 16.37°$, Lat: $\sim 48.2°$) and thus the WKT of a geographic point object representing Vienna using *EPSG* 4326 can be written as:

```
POINT(16.37 48.2)
```

To see if this point is inside the polygon representing the image PostGIS provides the function `Intersects(geometry,geometry);`. This functionality is performed by the Geometry Engine - Open Source[6] (GEOS) module, which is used by PostGIS for many functions in connection with object manipulation and comparison. The following SQL-command retrieves whether the image "eoc06182_20010216T091614" is actually showing Vienna or not:

```
SELECT Intersects(GeometryFromText('POLYGON((16.02063625 48.86594
772,16.26286888 48.89667965,16.81641535 47.30641733,16.58351857 4
7.27637656,16.02063625 48.86594772))',4326),GeometryFromText('POI
NT(16.37 48.2)',4326));
```

Not surprisingly, the result is: "`TRUE`" (compare figures 1.3 and 1.4). The applied function allows all possible geometries as input and returns `TRUE` if the geometries have any point in common and `FALSE` if they share no point at all. Thus this function can be used for all kinds of spatial searches.

2.3.3 Problems

The first problem occurs from the fact that Earth is not flat. PostGIS threats all objects as if they are projected onto a plain surface. Longitude

[6] URL: http://geos.refractions.net

is defined to be in the interval $lon : -180° \leq lon \leq 180°$. What happens if a search crosses the line of ±180° longitude, which lies opposite the prime meridian? This line is mostly identical with the international date line, which reveals a new problem if the search is also considering date. The worst case is caused by the poles. If one of the poles is covered by the image everything gets much more complicated.

PostGIS assumes all objects to lie in a plain surface using the coordinates in a two dimensional Cartesian coordinate system. On the horizontal axis, which is labeled x, the longitude is measured and on the vertical axis, labeled y, the latitude. Figure 2.2 shows Austria like PostGIS would "see" it. The

Figure 2.2: Austria using simple longitude and latitude (WGS84) coordinates in a Cartesian coordinate system.

picture shows also the outline (orange) of the same KOMPSAT image as used above. There is no problem with this image, as previously demonstrated. The problem occurs, when the image crosses the borders of a rectangle, defined with longitude (±180°) and latitude (±90°), like in figure 2.3. The image is enlarged

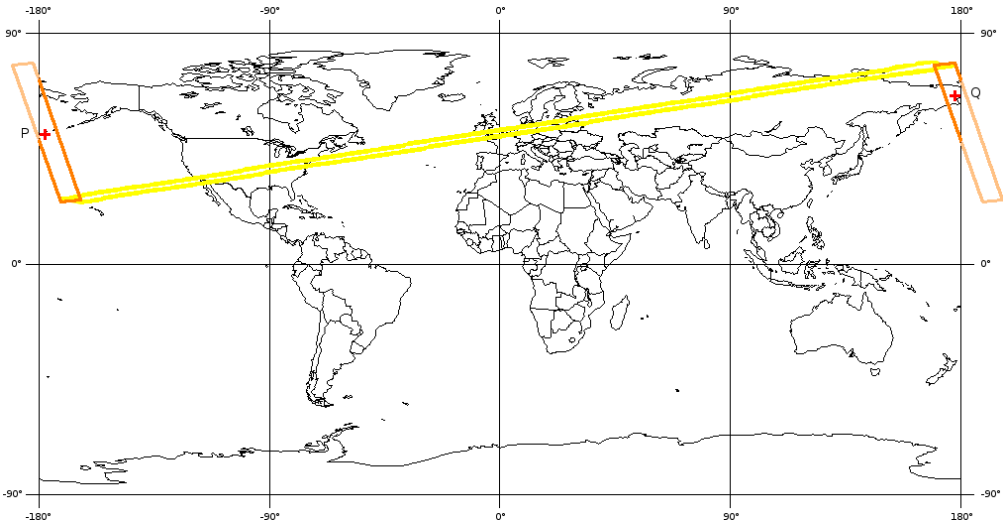


Figure 2.3: Image crossing the meridian of longitude ±180°.

to make the effect better visible. The correct outline is shown in dark orange whereas the light orange lines show the two possibilities how the image can be

stored in PostGIS if the definition of longitude is ignored. The yellow polygon shows a third possibility which does not ignore the definition but is completely wrong because nearly no point inside this polygon is in the image.

If the point $P = (-178°/50°)$ is shown by the image, the question could possibly be answered in the wrong way with: "False" if the alternative on the right (of the two ones shown in light orange) is stored in the database. It is necessary to ignore the definition of longitude and search for the point as $P = (182°/50°)$ to get the correct answer. The same mistake occurs when the point $Q = (178°/65°)$ is queried and the left alternative is stored. Obviously none of the possibilities fulfills the need for correct answers at all times and possibilities.

Fortunately, there is another extension to PostgreSQL to cope with these problems called pgSphere[7], which provides spherical data types, functions, and operators. The latest version is 1.0_Beta1 since the software is still under development. This extension adds new data types like spoint, sline, spoly, etc. For example a point can be defined by `spoint'(16.37d,48.2d)'` where "d" stands for input in degree. The advantage of this software reveals the definition of a point like `spoint'(16.37d,100d)'`. The wrong input of latitude 100° is recognized and the point is corrected to $P = (196.37°/80°)$.

pgSphere also provides some additional search functions. The operator "@@", for example, returns whether the objects overlap each other or not. This clearly points out that a combination of PostGIS and pgSphere can perform all tasks required. PostGIS is at least necessary for re-projections and pgSphere to cope with the 180° meridian and the poles.

The only problem, or maybe another advantage, with pgSphere is that it takes the Earth curvature into account and thus there are some differences in the search outputs compared to PostGIS. All objects are treated as they are lying on a sphere and lines are always drawn along great circles. The shortest path between two points on a sphere is always along a great circle. For small objects there is no identifiable difference to PostGIS, but as soon as the objects get bigger, so does the difference. In PostGIS all points on lines through two points of the same latitude have this latitude. Whereas in pgSphere they do not have the same latitude as can be seen by the orange line in figure 2.4. Points on the equator are the only exception because the equator is a great circle. It is up to the programmer to decide which search is the appropriate onespecific problem.

There is also a constraint in pgSphere for polygons, namely the maximum dimension of a polygon must be less than 180°. This is essential because two points in pgSphere are connected along the shorter part of the great circle through both points and not the longer one. There is generally no problem since the images are usually small enough. KOMPSAT-1 for example can capture images with a maximum length of $\sim 800\ km$ which corresponds to a difference

[7] URL: http://www.pgastro.org/cgi-bin/wiki.pl?pgSphere

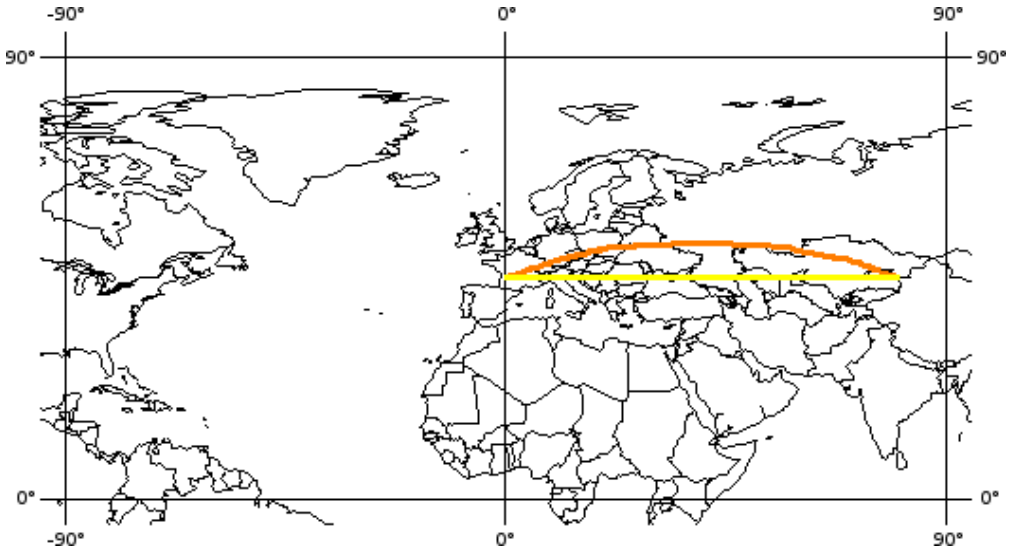


Figure 2.4: Lines in PostGIS (yellow) and pgSphere (orange).

in latitude of about $\sim 7.2°$. It is necessary to remember that pgSphere uses a sphere, not a spheroid, for object representation and thus a change of one degree latitude corresponds to the same difference in kilometers, $\sim 111\ km$ everywhere on the sphere. Unlike for latitude, for longitude this does not work, since meridians, lines of constant longitude, converge at the poles and are furthest away from each other at the equator where one degree accords to $\sim 111\ km$.

It depends on the application which extensions are best to be used. At ARC-sr at the moment, just PostGIS is being used because there are only images of Europe and nearby countries to store and therefore, no problems with the 180° meridian and the poles arise. The following section explains the technical details of the database design implemented at ARC-sr.

2.4 ARCS SATdata Catalog

The metadata catalog service implemented at ARC-sr, called "ARCS SATdata Catalog", is made up of the PostgreSQL database management system with PostGIS extension, some PL/pgSQL functions for data manipulation, an administration tool, programmed in PHP and accessible with every standard web browser, the file system on hard disc where all the images and tiles are stored, and the possibility to explore and search the service via different access points.

2.4.1 Database Design

2.4.1.1 Logical Design

The logical schema of the database is best explained in a graphic. Thus, the schema is shown using diagrams of the Unified Modeling Language (UML) (Figure 2.5) and also the Extended Entity-Relationship (EER) model (Figure 2.6) as described by Kemper and Eickler (2001); Teorey, Yang, and Fry (1986). UML is a non-proprietary, third generation modeling and specification language, used especially for object-oriented modeling. The UML class diagram and the EER diagram are both for high-level descriptions of logical data models, and they provide a graphical notation for representing such data models.

In the UML class diagram (Figure 2.5) there is one rectangle for every entity (class) with a name, e.g. "image_metadata", attributes with type, e.g. "ImageName" with type `CHAR(24)`, and optional operations and methods. The " + " before attributes signals that they are publicly visible whereas a " − " means that they are private. The classes can be connected. Arrows associated with connection lines signal the direction in which associated objects could be determined, e.g. between the entities "image_metadata" and "City" it could be determined which image "is over" which city. The numbers or asterisks are showing how many objects are related, e.g. an image is over at least one city (1..∗) whereas a city can be shown by any number of images (∗). The black rhombus stands for a composition which means that objects on the side of the rhombus are higher-ranked and that lower objects have to be associated with exactly one higher object, e.g. one sensor is exactly mounted on one platform. The last graphical notation to explain are the big arrows next to the "Process" class. This is a generalization which means that one process must either be of type "raw", "geo", "ort", or "wms". The light gray process type "urb" does not exist at the moment but could be introduced in future.

In the middle of both diagrams the "main"-entity called "image_metadata" is shown in light orange. For every image at ARC-sr an entry in this entity is created. One or more columns in one table, that can be used to (uniquely) identify a row in a table are called a (unique) key. One of the unique keys is the preferred way to refer to rows and is defined as the table's primary key. The primary key of this table is called "ImageName" and is a string with a fixed count of 24 characters.

The naming convention for the attribute "ImageName" is: Sensor (3 characters, "eoc" or "msc"), Orbit (5 digits), Underscore, Date (8 characters in year-year-year-year-month-month-day-day format), T, Starting time (6 characters like hour-hour-minute-minute-second-second). The date and time notation is conform to ISO (ISO 8601[8]). Accordingly can be seen, that the example from above "eoc06182_20010216T091614" was captured at orbit number 06182 on February 16, 2001 starting at 9:16:14 Coordinated Universal Time (UTC)

[8] URL: http://www.iso.org/iso/en/prods-services/popstds/datesandtime.html

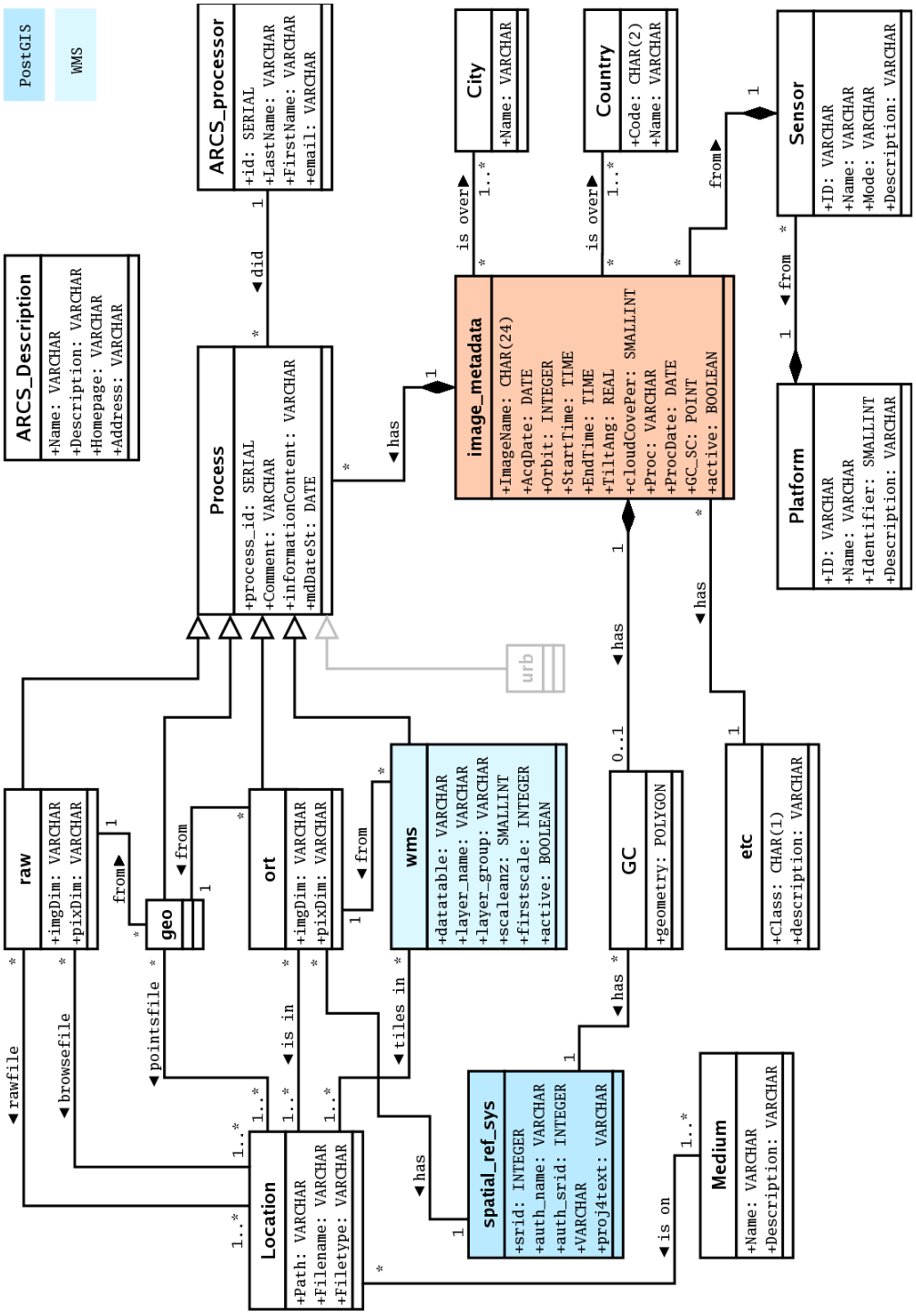


Figure 2.5: UML class diagram as described by Kemper and Eickler (2001) for the database at ARC-sr

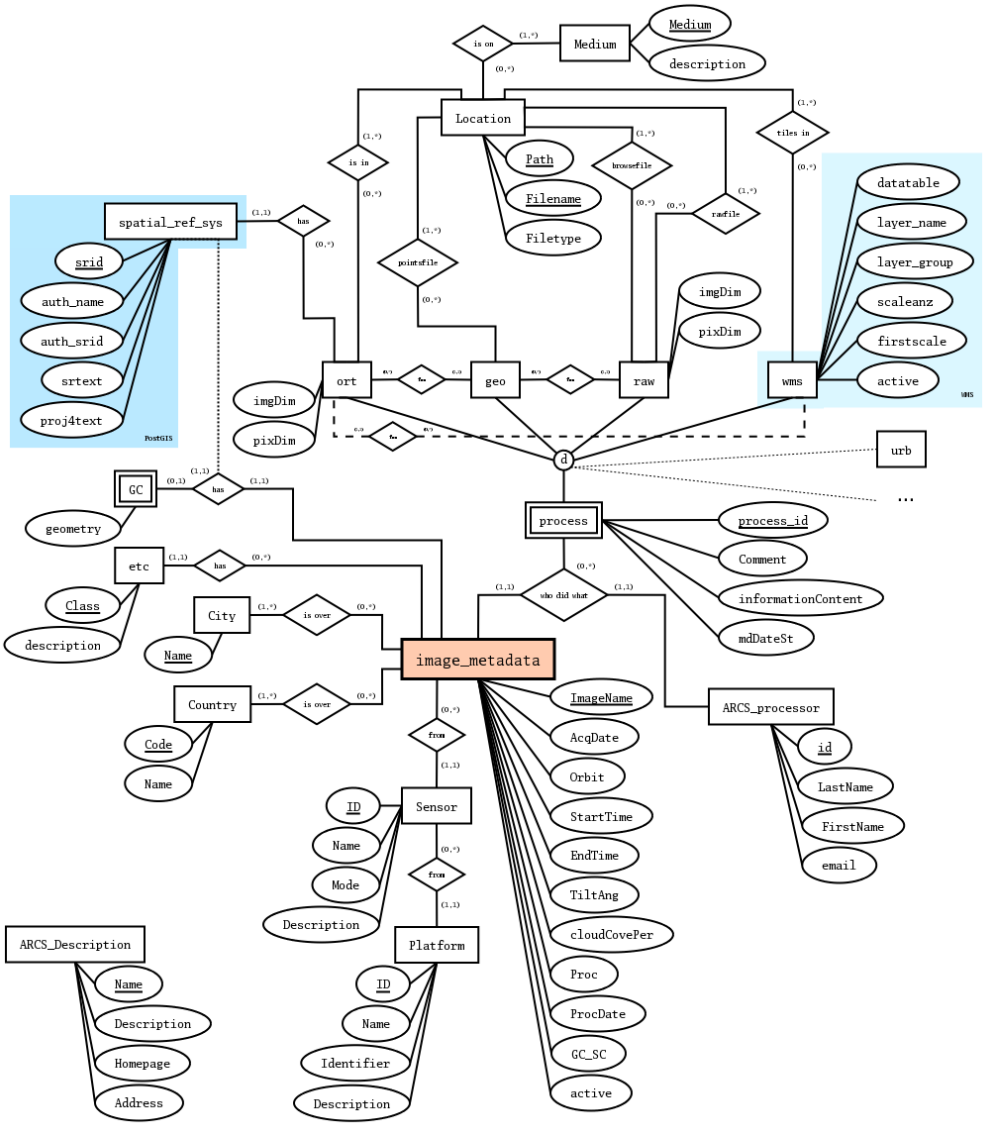


Figure 2.6: EER diagram as described by Teorey et al. (1986); Kemper and Eickler (2001) for the database at ARC-sr

with KOMPSAT-1's EOC. The other attributes like "AcqDate" (date of acquisition), "Orbit", "StartTime", etc. are shown in the diagrams.

The second basic entity is "process" which is a generalization of "raw", "geo", "ort", "wms", and possible future processes e.g. "urb". Generalization means, that the generalized entities inherit the superclass's, here "process" attributes. Thus, every attribute which is common to all processes, like "comment", is stored in the table "process". Every processing step done with an image gets logged here. "raw" implies the process of getting a new raw-image and storing it in the file system together with a browse image (quick-look) which needs to be generated as well. "geo" describes the process of looking up

geographic coordinates and assigning them to image-pixels. "ort" is the process of taking the points-files from the "geo" process, orthorectifying the image, and storing it in the file-system. "wms" (light-blue) finally is the process of producing tiles resulting of the "ort" process to be used for the OpenGIS® services. Further explanations of these processes are provided in section 2.4.4.

Hence, it is very important to store the location of all these files in the file system. This is achieved by the relationships between the entity "Location" and the processes. We need to store the location of the raw-image, the browse-image, the points-file, the ort-file, and the tiles. To make a backup handling as easy as possible the entity "Medium", which stores which locations exist on which media e.g. "HD" for hard-disc, is added to the schema.

The table `spatial_ref_sys` (blue) is the previously presented PostGIS internal table storing all projection information. This table is also used to describe the projection system applied to the orthorectified images and tiles. This is very important for on-the-fly projection transformations within the OpenGIS® services as well as for data delivery.

2.4.1.2 Physical Design

The physical schema of the database is given below. It results out of the translation of the diagrams from the logical schema into relations also referred to as tables. Table names are typed bold, primary keys underlined, and column types uppercase. Attributes are written in curly braces. The dot in attribute names represents a reference, e.g. "image_metadata.ImageName" is a reference to the column "ImageName" of the table "image_metadata".

ARCS_Description{Name: VARCHAR, Description: VARCHAR, Homepage: VARCHAR, ADDRESS: VARCHAR}
Platform{ID: VARCHAR, Name: VARCHAR, Identifier: SMALLINT, Description: VARCHAR}
Sensor{ID: VARCHAR, Name: VARCHAR, Mode: VARCHAR, Description: VARCHAR, Platform.Name}
etc{Class: CHAR(1), description: VARCHAR}
image_metadata{ImageName: CHAR(24), AcqDate: DATE, Orbit: INTEGER, StartTime: TIME, EndTime: TIME, TiltAng: REAL, cloudCovePer: SMALLINT, Proc: VARCHAR, ProcDate: DATE, active: BOOLEAN, etc.Class, Sensor.ID, GC_SC: POINT}
City{Name: VARCHAR}
is_over_City{image_metadata.ImageName, City.Name}
Country{Code: CHAR(2), Name: VARCHAR}
is_over_Country{image_metadata.ImageName, Country.Code}
Location{Path: VARCHAR, Filename: VARCHAR, Filetype: VARCHAR}
Medium{Medium: VARCHAR, Description: VARCHAR}

Location_is_on_Medium{Medium.Medium, Location.Path, Location.Filename}
GC{image_metadata.ImageName, geometry: POLYGON}
ARCS_processor{id: seriel, LastName: VARCHAR, FirstName: VARCHAR, email: VARCHAR}
process{process_id: seriel, image_metadata.ImageName, ARCS_processor.id, Comment: VARCHAR, informationContent: VARCHAR, mdDateSt: DATE}
raw{process.process_id, process.image_metadata.ImageName, imgDim: VARCHAR, pixDim: VARCHAR}
rawfile_Location{raw.process.process_id, raw.process.image_metadata.ImageName, Location.Path, Location.Filename}
browse_Location{raw.process.process_id, raw.process.image_metadata.ImageName, Location.Path, Location.Filename}
geo{process.process_id, process.image_metadata.ImageName (=raw.process.image_metadata.ImageName), raw.process.process_id}
pointsfile_Location{geo.process.process_id, geo.process.image_metadata.ImageName, Location.Path, Location.Filename}
ort{process.process_id, process.image_metadata.ImageName (=geo.process.image_metadata.ImageName), geo.process.process_id, imgDim: VARCHAR, pixDim: VARCHAR, spatial_ref_sys.srid}
ortfile_Location{ort.process.process_id, ort.process.image_metadata.ImageName, Location.Path, Location.Filename}
wms{process.process_id, process.image_metadata.ImageName (=ort.process.image_metadata.ImageName), ort.process.process_id, datatable: VARCHAR, layer_name: VARCHAR, layer_group: VARCHAR, scaleanz: SMALLINT, firstscale: INTEGER, active: BOOLEAN}
tiles_Location{wms.process.process_id, wms.process.image_metadata.ImageName, Location.Path, Location.Filename}
spatial_ref_sys{srid: INTEGER, auth_name: VARCHAR, auth_srid: INTEGER, srtext: VARCHAR, proj4text: VARCHAR}

The actual definition of the tables can be explored in the appendix, A.2. These definitions are written in SQL. Lines starting with "–" are comments and are to be ignored. In order to use this script a database with PostGIS support is necessary. Further to create one the terminal-based front-end to PostgreSQL "psql" is required. If PostgreSQL is installed properly it can be entered by typing the command "psql" followed by an already created database on the

command line (every standard installation of PostgreSQL creates the database "template1").

With the command

```
CREATE TABLESPACE ksat_postgis LOCATION '/home/ksat/postgis';
```

a storage place for the catalog ksat_postgis is created. The location needs to be an existing, empty directory that is owned by the PostgreSQL system user (normally "postgres").

To create a database called "ksat" the command

```
CREATE DATABASE ksat TABLESPACE=ksat_postgis ENCODING='SQL_ASCII';
```

is used. The character encoding is set to "SQL_ASCII" to get correct characters. Now the only thing left is to add PostGIS support to the new database. If PostGIS is installed properly this is done with the following two commands

```
CREATELANG plpgsql ksat
psql -f /usr/local/pgsql/share/contrib/lwpostgis.sql -d ksat
```

typed in the command-line (not the psql tool!). Note, the directory of the "lwpostgis.sql" script depends on the installation. Finally the script shown in the appendix can be used to create the tables by typing

```
psql -f ARCS_SATdata_Catalog.sql ksat
```

2.4.2 PL/pgSQL Functions

To complete the database design some PL/pgSQL functions were developed to make the data manipulation easier. PL/pgSQL is a loadable procedural language for the PostgreSQL database system. The script is shown in the appendix A.3. In order to load it into the database the following command needs to be executed on the command line.

```
psql -f ARCS_SATdata_Catalog_Functions.sql ksat
```

This adds the functions

```
insert_process(VARCHAR,INTEGER,VARCHAR,VARCHAR)
insert_raw(VARCHAR,INTEGER,VARCHAR,VARCHAR,VARCHAR,VARCHAR)
insert_geo(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER)
```

```
insert_pointsfile(VARCHAR,INTEGER,VARCHAR,VARCHAR,VARCHAR,VARCHAR)
insert_ort(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER,VARCHAR,
     VARCHAR,INTEGER)
insert_wms(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER,VARCHAR,
    VARCHAR,VARCHAR,INTEGER,INTEGER,BOOLEAN)
select_countries(varchar)
select_cities(varchar)
```

to the database.

The function `insert_process();` inserts just the values which are common to all processes. This function is further used by the functions `insert_raw();`, `insert_geo();`, `insert_ort();`, and `insert_wms();`, which insert processes according to their names. The last two functions `select_countries();` and `select_cities();` are used to collect all countries or cities an image is showing into a single string. `insert_pointsfile();` inserts the location of a new points-file. For detailed information on these functions see appendix A.3.

2.4.3 Administration Tool

For easier, controlled, and advice providing content management an administration tool was developed. This tool can be accessed with every standard web browser e.g. like Mozilla's Firefox. Currently it provides functions for data-input and data-viewing. It is intended to develop the tool further with functions for data-update and data-deletion.

The tool is written with PHP and JavaScript. PHP is used for all the server sided tasks like data-retrieval and manipulation in the database. The PL/pgSQL functions discussed in section 2.4.2 are used through PHP. The JavaScript code is executed at the client side within the browser mainly to prevent the user from entering wrong or incomplete data. To customize the administration tool the variables in the file "settings.php" can be changed. The most important ones are those that start with "db" and control the connection to the database. How PHP connects to the database will be explained in section 4.4 and the JavaScript functionality in section 4.5.

After entering the tool by providing username, password, and database (name of database to edit/view) a page providing some overall information like the current date, build date and number, login-information, and links to the special tasks pages is shown (Figure 2.7). It provides the possibilities to insert miscellaneous data like a new processor (operator), city, or medium, to insert an image, to insert processes, to view already stored data, as well as to logout.

The selection of "Insert image data" results in a page like the one in figure 2.8. It requests all necessary information for the storage of a new image. The select boxes for cities and countries allow multiple selections if the image shows

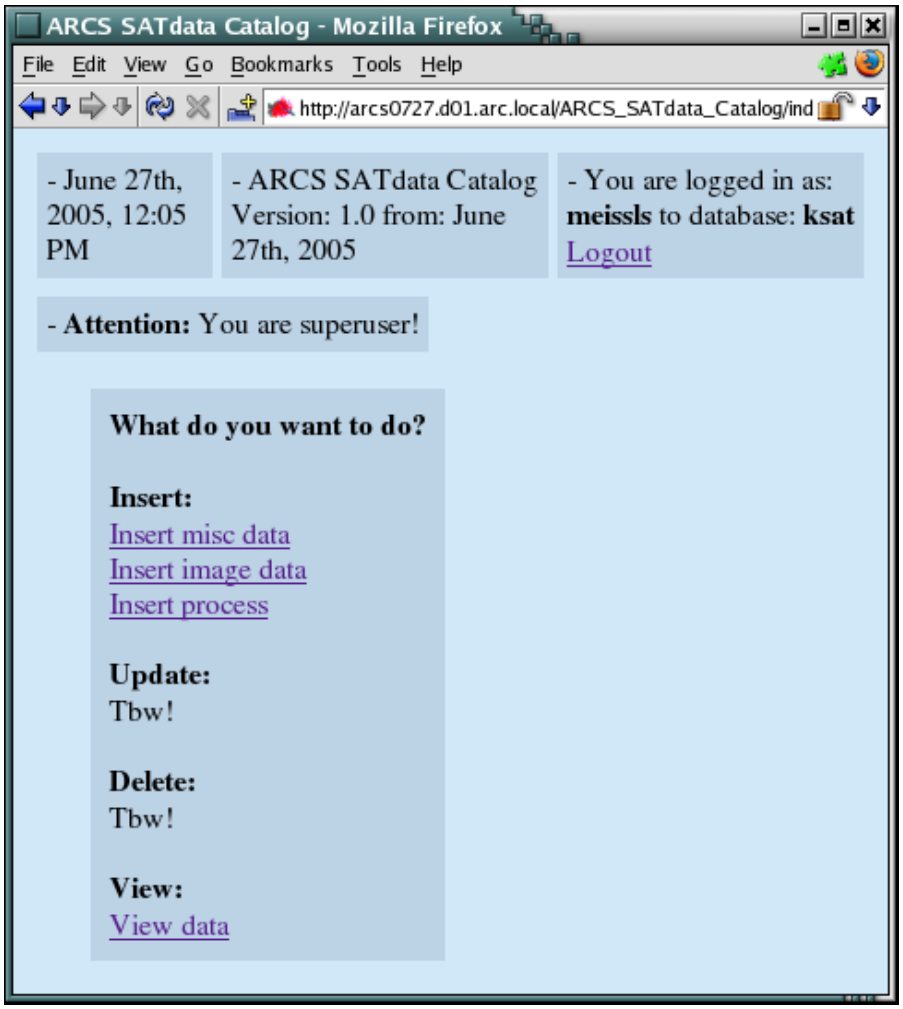


Figure 2.7: Administration tool: Task selection

more than one of them. After choosing one of the buttons labeled "INSERT" the entered information is validated by JavaScript routines and (if everything is correct) sent to the server. In the next step the sent information is stored in the database on the server and the user is redirected to a new page for entering processes.

There are two ways of getting to the window in figure 2.9, either after the successful insertion of a new image or with the link called "Insert process". A unique form for each possible process is shown and awaits the input of the corresponding information. The processes and the associated information are explained in detail in section 2.4.4.

Furthermore, there is the possibility to show already stored information by selecting "View data". By selecting the desired image the user gets all the information stored along with this image in one page.

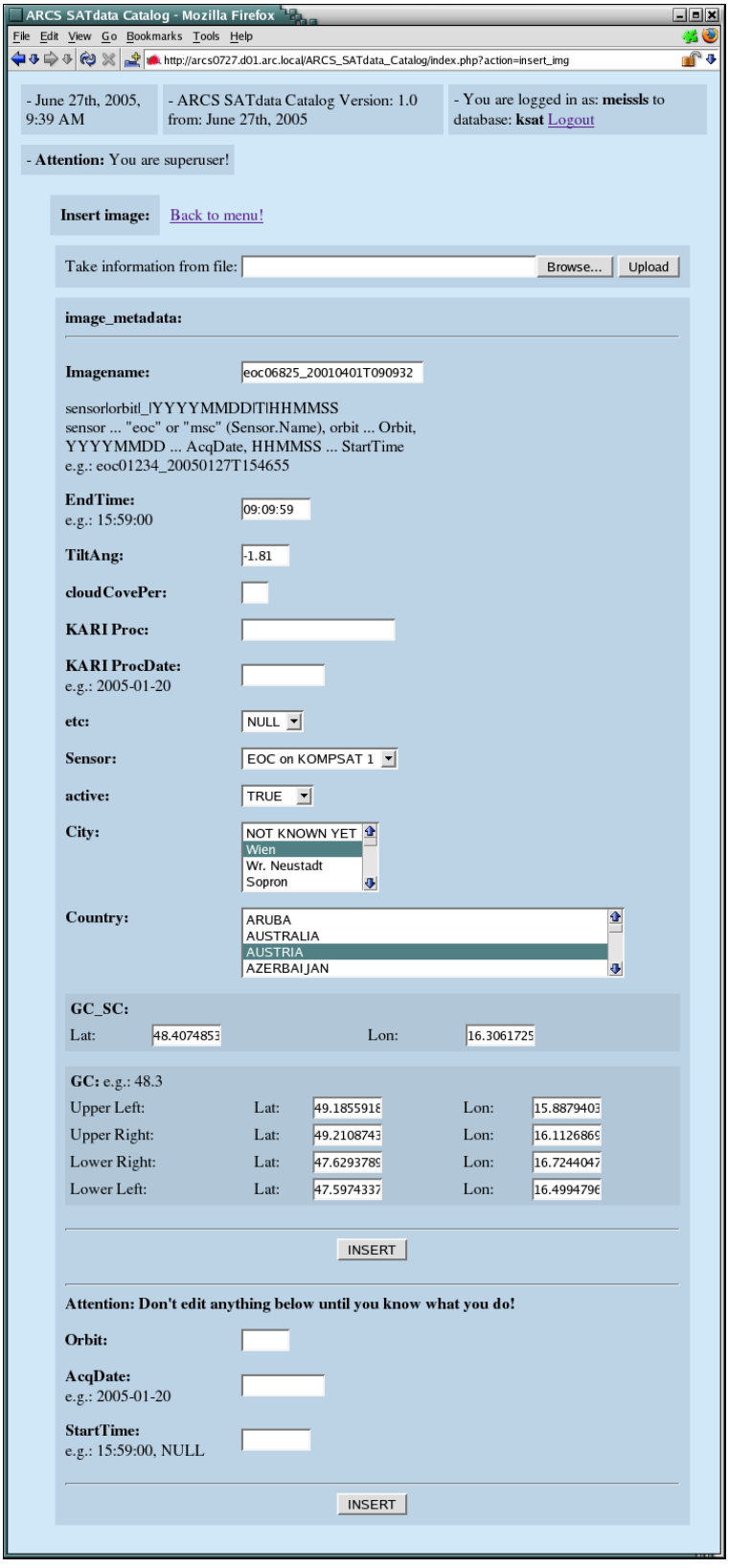


Figure 2.8: Administration tool: Insert image form

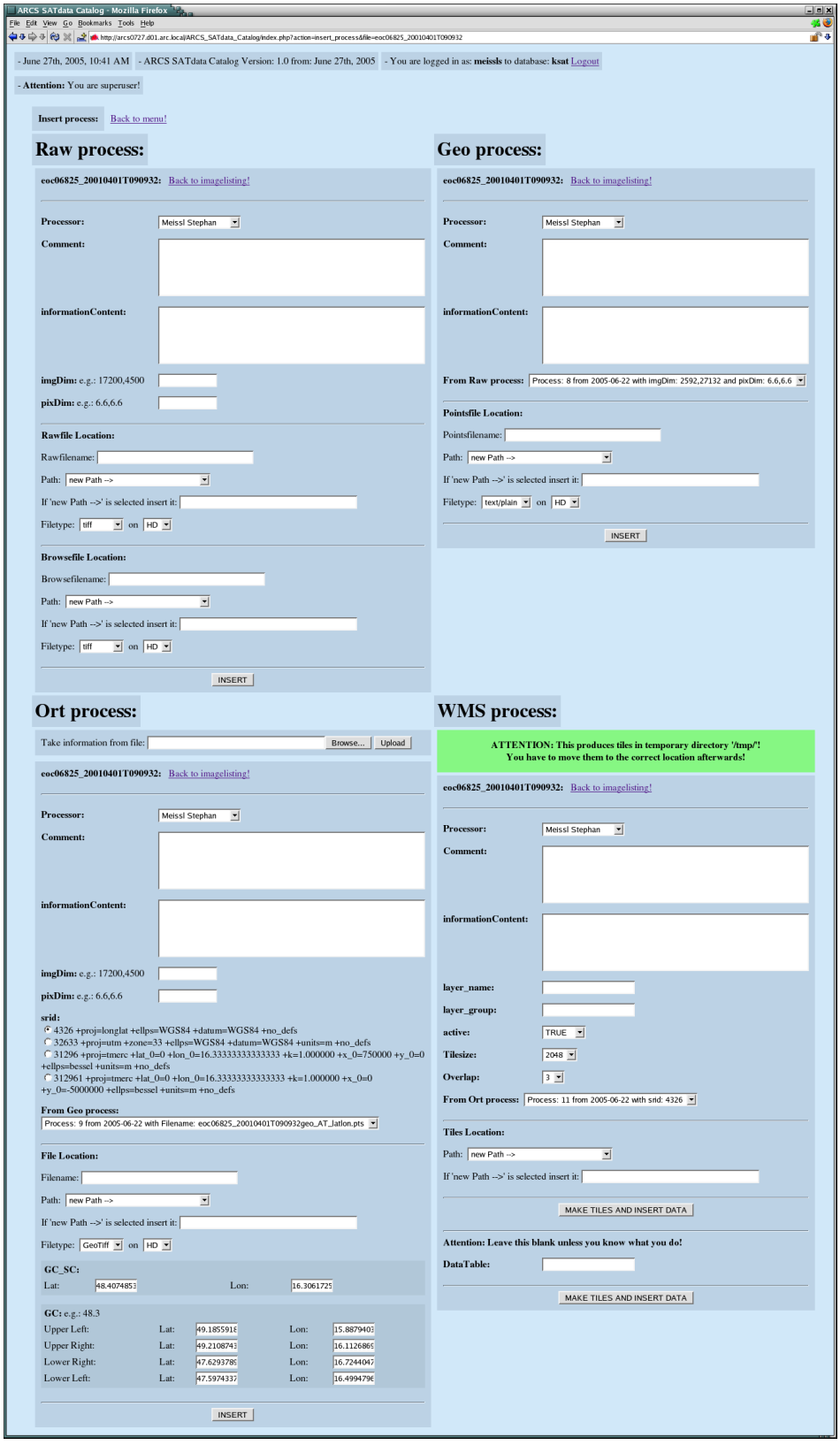


Figure 2.9: Administration tool: Insert process form

On top of the page there is always some overall information shown like the current date, build date and number, and login-information. Below, information which task is selected is provided. There is always the possibility to go back to the initial menu (figure 2.7) by following the link "Back to menu!".

2.4.4 Description of Processes and Workflow

The names of the processes identified and defined at ARC-sr so far are: raw, geo, ort, and wms. These definitions are very special and are maybe only valid for ARC-sr. There is the possibility of creating new processes like urb, standing for an urban product like land-use classification.

The raw-process is the process of storing a new image in the file system. A unique filename according to the conventions introduced in section 2.4.1 and extended as follows is created. The extensions are: the first 24 characters are followed by the string "raw_" to describe the file contents, subsequently a two character (uppercase) country code (following ISO 3166[9]) of the country where most of the image is within is added, and the last part is a three character file format extension. Thus, the former example would look like "eoc06182_20010216T091614raw_AT.tif". In addition a browse image is created and named after the same convention, only "raw" is changed to "brs", e.g. "eoc06182_20010216T091614brs_AT.png". All undertaken steps and locations of created files need to be stored in the database. Figure 2.9 shows which inputs are possible for a raw-process. The inputs for "imgDim", "pixDim", "Rawfile Location", and "Browsefile Location" are mandatory whereas "Comment" and "InformationContent" are optional.

The process of creating a points-file is called "geo". A points-file is a plaintext-file storing a header and pairs of geographic coordinates and pixel locations, known as GCP, which are used to orthorectify an image. A typical filename would be "eoc06182_20010216T091614geo_AT_latlon.pts" where the string "latlon" refers to the applied projection, in the example the geographic coordinates which are (simple) latitude/longitude coordinates. One line of this file could be written as (the value in the middle is the elevation)

```
16.4191 48.2065 159.040000 1676.000000 11597.000000
```

This format is dependant on the software used for satellite image processing. At ARC-sr the tool ENVI (RSI) is used.

After the creation of a points-file the sting "raw" in the according raw-file gets renamed to "geo" (e.g. "eoc06182_20010216T091614geo_AT.tif") to show the existence of a points-file. The form in the administration tool needs no more information than the points-filename however still allows some optional

[9] URL: http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/-index.html

inputs. If there are two or more raw-processes for this image the user needs to select the one the points-file was created for. The presence of more than one raw-process could occur if the size of the image is to big and gets split up into some smaller files and therefore a new raw-process is created for every new piece of the image. However most likely there exists only one raw-process because even if an image gets split up the smaller images will be treated as independent images.

The "ort" process is a very important one because afterward the corner coordinates of the image are finally known. Thus, from now onwards there is the possibility to perform a spatial search in the archive. The process's name derives from orthorectification and this is what the process intends to describe. The input to the database for this process has to include a new "imgDim", a "srid" (the geographic projection system identifier), the "File Location", the coordinates of the scene center ("GC_SC"), and all corner coordinates ("GC").

The last process is called "wms". This process is significant for the following chapters and thus is described separately in the following section (2.4.5).

2.4.5 The WMS Process

"wms" is the name of the process to prepare the image for the usage within the WebMap-, and WebCoverageService which will be introduced in chapter 3. A short response time is very important for these services. To increase speed and performance the large raster images are tiled into smaller pieces and shapes are created for their outline. These shapes are saved in the database. After the database has been searched for the required area (based on the outline vector shapes) just the actually required raster images have to be loaded and processed. The tilling has the additional advantage that only tiles that are not black need to be saved. Figure 2.10 shows an example. The blue lines are the shapes of the tiles. It can be seen, that there are no tiles in the lower left and in the upper right corners where only black areas occur.

A script which performs the tiling (see appendix A.4) needs some parameters. The parameters like "username", "database", etc. are known because of the login. Other parameters like "layer_name", "tilesize", "overlap", etc. need to be specified. "layer_name" is used for the WMS and WCS to produce human readable names for the different layers. "layer_group" can be used to sort the layers into different groups (further explanation see 3.6).

Another parameter is called "tilesize". This parameter tells the script how big the output tiles should be (in pixels). A good choice for this size would be 2048 $pixels$. This is just one number since the tiles are intended to be squares. To prevent possible white lines between tiles they overlap, usually by 3 pixels, but this default value can be changed. White lines could occur if the images are reprojected and thus distorted. The size of the example image is 13380 x 27240 $pixels$ and is therefore tiled into 7 x 14 tiles because $7 = \lceil \frac{13380-3}{2048-3} \rceil$

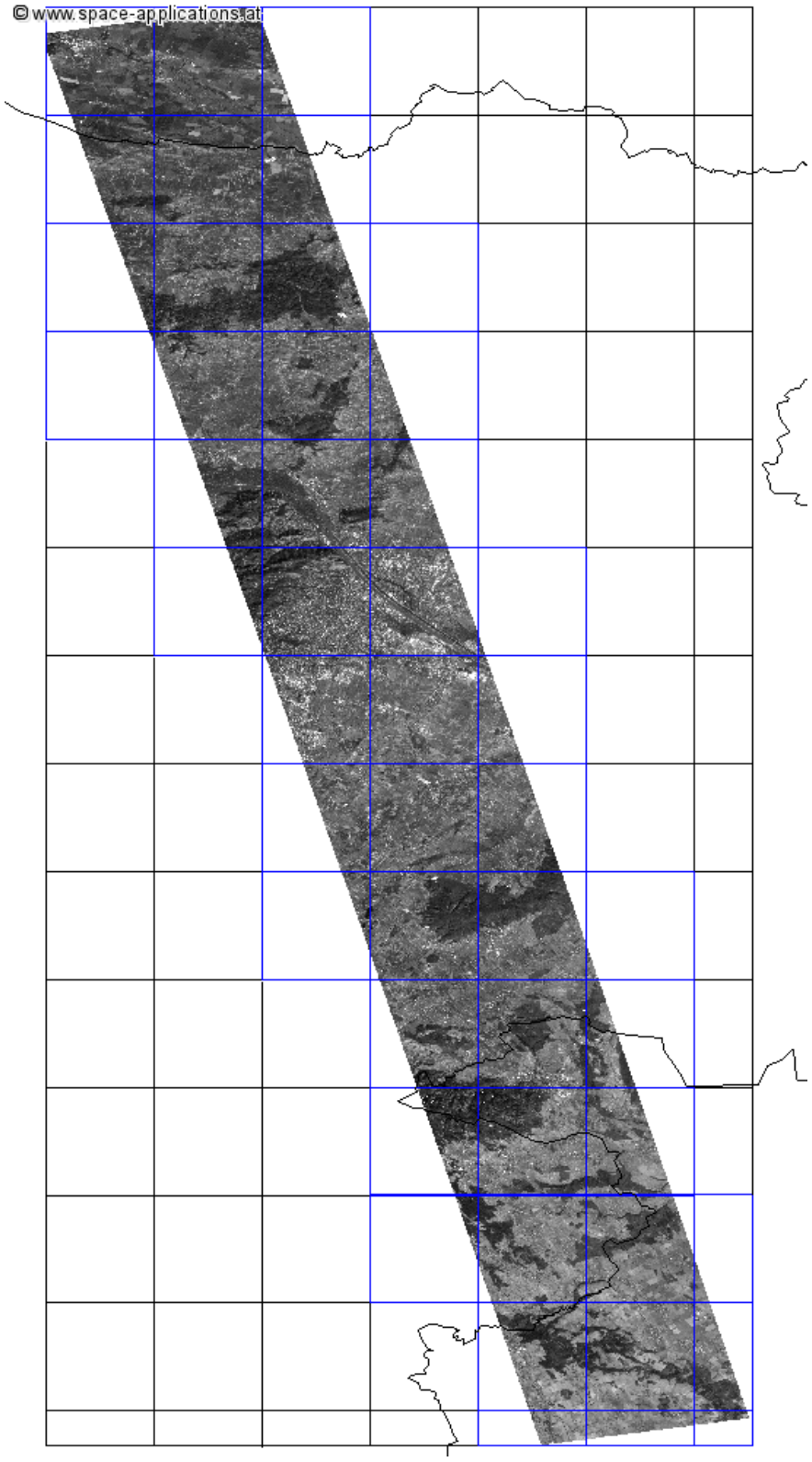


Figure 2.10: Tiling of image. Blue lines are outlines of tiles. Black lines would represent the full image.

and $14 = \lceil \frac{27240-3}{2048-3} \rceil$ (where $\lceil \rceil$ is the mathematical ceiling function which returns the next greater integer). Only 47, out of the 98 tiles created, need to be saved since the other contain no data (black areas).

Finally, the script needs to know the location where the tiles will be stored to enter this information in the database. Unfortunately, it is not possible to let the script produce the tiles in the correct directory due to permission issues. Thus, it is necessary to move the files manually to the correct location and adjust permissions when the script has finished.

The tiling process also generates so-called "overviews" which are images at half resolution in both dimensions. This means that four pixels always get combined to one pixel in the overview, which is therefore only a quarter in size of the original image. These overviews are also tiled and the process continues until the smaller image extent is shown by one tile. Therefore there is one dimension where the overview is not tiled any more. In the previous example there are 7 x 14 tiles after the tiling. The first overview has 4 x 7 tiles because 4096 x 4096 *pixels* are shrunk to 2048 x 2048 *pixels*, the next one has 2 x 4 tiles (8192 *pixels* shrunk to 2048 *pixels*), and the last one has 1 x 2 tiles (16384 is bigger than 13380). This overview has only one tile in one direction and the building of overviews stops. Because of the black areas only 71 tiles out of the 136 created are saved.

The tiles are named according to the following convention: ImageName (in lowercase), the string "_scale", a scale-number starting with 1, "_", the starting pixel (upper left corner of the source window) in x direction, "_", the starting pixel in y direction, and the file extension ".tif". An example would be: "eoc06182_20010216t091614_scale1_0_2045.tif".

In the database a table for every image is created and named after the ImageName but only in lowercase which is required for the WMS software used at ARC-sr. This table name can be changed with the field "datatable". The permission to query and retrieve (select) data from this table is granted to the user specified in the settings file (see end of section 2.4.3). The script also calculates a scale-factor for the first initial tiling which is also required by the WMS software.

This tiling and building of overviews has turned out to be the fastest solution among some alternatives (e.g. just GeoTIFF format, GeoTIFF with the built-in tiling and overview options, some compressions like JPEG) which were all tested at ARC-sr. On computers where the Central Processing Unit (CPU) is much faster than the hard disc it would be better to compress the tiles but normally compression decreases speed. It is also recommended to store the images in the most often used projection because reprojecting images on the fly is very time consuming.

After discussing all topics regarding data storage, date manipulation, local data search, and retrieval, I like to analyze how this data can be accessed in

various sophisticated ways. How can the data be searched from remote devices via a network? How can the date be viewed, and how can the data be ordered and obtained? The following chapter introduces the OpenGIS® services WMS, WFS, and WCS which are primarily used to view data interactively. It will be discussed how they are specified, how they work, which software is available, and how it is used. The following chapter will cover the client side of these services by introducing the client developed at ARC-sr.

Chapter 3

WebMap, WebFeature and WebCoverageService

The previous chapter explained the needs of a spatial database and the usage with respect to satellite derived images. It explained which data needs to be stored and how data can be locally manipulated, searched, and retrieved. The software, design, and some tools which are used at ARC-sr were introduced. This chapter explains which and how services can make use of this database and knowledge about it. Especially the remote access via a network to these services is of interest.

This chapter starts with an introduction of the OpenGIS® web services WebMapService (WMS), WebFeatureService (WFS), and WebCoverageService (WCS). The service standards and specifications are explained as well as the software, the configuration and usage of this software implemented at ARC-sr. After reading this chapter the reader will understand these services and be able to set up a computer running these services.

3.1 The Open Geospatial Consortium

The services have been specified by the Open Geospatial Consortium, Inc.[1] (OGC). OGC is an international consortium of companies, agencies and universities. They define themselves as "a non-profit, international, voluntary consensus standards organization that is leading the development of standards for geospatial and location based services in different working groups". Through their member-driven consensus programs, they work together with government, private industry, and academia. Virtually all well-known providers of software for Geographic Information Systems (GIS) take part in the OGC. GIS is mostly defined as software systems for managing spatial data and associated attributes.

[1] URL: http://www.opengeospatial.org

It is capable of integrating, storing, editing, analyzing, and displaying this type of data.

OGC was founded by eight members in 1994 and has grown to over 280 members worldwide today. The vision of OGC is "A world in which everyone benefits from the use of geospatial information and supporting technology.". Their mission is "To lead the global development, promotion and harmonization of open standards and architectures that enable the integration of geospatial data and services into user applications and advance the formation of related market opportunities.". OGC tries to accomplish this mission by formalizing OpenGIS specifications through consensus, organizing interoperability projects (e.g. testbeds), promoting demand for interoperable products, etc.

They try to fulfill several user needs like the maximization of investments in geoprocessing systems and data, the sharing and reuse of data, the selection of the best tool for a certain job, and the easy use of geospatial data in more applications for less trained people. They also aim to make geospatial data and services available over different networks, programs, and platforms through clearly specified open and extensible software application programming interfaces for GIS and other mainstream technologies.

OGC focuses on open web based interface specifications which allow software vendors to implement their products using interoperable interfaces and provide end-users a larger pool of interoperable web based tools for geodata access and related geoprocessing services. Geospatial interoperability is defined as the ability for two different software systems to interact with geospatial information mostly via the Internet.

A web service is defined as any software that makes itself available over the Internet, that supports interoperable machine-to-machine interaction over a network. Software applications written in various programming languages and running on various platforms use web services to exchange data over computer networks, e.g. the Internet, in a similar manner as inter-process communication on a single computer. This interoperability is possible due to the use of open standards, like those of OGC.

3.2 WebMapService Specification

The WMS is one of OGC's most popular standards. This service produces maps, which are visual representations of georeferenced data. These maps are not the data itself. The specification defines the syntax of requests as well as the format and features of the result.

The specification is available in multiple versions. The latest version is 1.3 (OGC IS, WMS 1.3, 2004)[2] from August 2, 2004 which is equivalent to the

[2]URL: http://portal.opengeospatial.org/files/?artifact_id=5316

draft version of the ISO/DIS 19128 "Geographic information – Web map server interface". In this thesis, the version 1.1.1 (OGC IS, WMS 1.1.1, 2002)[3] has been used because it is the latest version supported by the software MapServer used at ARC-sr. This software tool will be introduced in section 3.6.

A normal WMS can answer three different operations, where the first two are required for every WMS and the last is optional. The behavior of a WMS can be extended by the Styled Layer Descriptor (SLD) which will be explained in section 3.7.2.

- GetCapabilities (required)
- GetMap (required)
- GetFeatureInfo (optional)

These operations can be invoked via every standard web browser through the World Wide Web (WWW) using Uniform Resource Locators (URL). An URL is a standardized address name layout for resources (such as documents or images) on the Internet. Detailed explanation will be given in section 3.5. The WWW is often mistakenly used as a synonym for the Internet, but the Web is actually a service that operates over the Internet. It is the service to provide an information space where items of interest, referred to as resources, are identified by URLs. The platform supported by the service is the WWW or more specifically Internet hosts implementing the Hypertext Transfer Protocol[4] (HTTP).

The service takes and processes the request, and delivers the according result:

- GetCapabilities requests are responded with a XML document describing the service-level metadata of operations and the content available at a actual implementation of a service.
- GetMap returns a georeferenced raster-image called a map with well-defined parameters.
- GetFeatureInfo returns information about particular features shown on a map.

The Extensible Markup Language[5] (XML) is a W3C-recommended general-purpose markup language. A markup language combines text and extra information about the text. Its primary purpose is to facilitate the sharing of data across different systems, particularly systems connected via the Internet. The

[3] URL: http://www.opengeospatial.org/docs/01-068r3.pdf
[4] URL to specification: http://www.ietf.org/rfc/rfc2616.txt
[5] URL to specification: http://www.w3.org/TR/2004/REC-xml-20040204/

World Wide Web Consortium[6] (W3C) is a consortium creating the software standards ("recommendations", as W3C calls them) for the WWW.

The parameters to be used when querying a WMS via URLs are well-defined and standardized (see section 3.7). Thus, requests do not need to take care of proprietary peculiarities of different servers. There is a clear differentiation between request, data storage, and data manipulation. A client can request individual map layers (sets of information to be shown) from different servers and combine them to a composite map. A particular WMS provider only needs to make the own data collection available and does not need to gather all data, accessible by the interface in one place.

An interesting and important feature of this independence is the Cascading Map Server. This is a WMS that behaves like a client for other WMSes and like a WMS to other clients. A Cascading Map Server can combine the information of several different WMSes into one service. It can also be used for data conversion like changing the output format or transforming the coordinate reference system.

In order to allow clients and Cascading Map Servers to communicate automatically with WMS servers the operations have to be specified precisely and strictly. The operations can be described as follows:

3.2.1 GetCapabilities

The GetCapabilities request returns a XML document. Since each WMS is independent this document has to represent a machine-readable description of its capabilities, of operations, and contents available at the WMS. This document has to be valid according to a XML Document Type Definition also available in the specification and online[7].

The data in the returned capabilities document includes some overall information like responsible party, layer descriptions, supported projection systems, viewable geographic extent, and the URLs to the supported requests. With this document clients are able to formulate valid requests to the WMS. It is further possible to construct catalogs of WMSes with this document.

3.2.2 GetMap

A client can request a distinct map with the GetMap request. The URL requested has to include the parameters describing which information should be shown on the map (layers), what portion of the Earth is to be mapped (a BoundingBox), the projection system, the output format, the output size, and

[6] URL: http://www.w3.org/
[7] URL: http://www.digitalearth.gov/wmt/xml/capabilities_1_1_1.dtd

often some more. Whether parameters and values are valid can be determined with the capabilities document. The WMS generally renders the required information in a raster-format such as PNG, GIF or JPEG.

3.2.3 GetFeatureInfo

The request GetFeatureInfo is optional. If a WMS allows this operation its maps are queryable. A client can request information about features on a map by adding URL additional parameters to the map. These parameters include a point geometry and the layers to be queried. The result is mostly returned as a HyperText Markup Language[8] (HTML) page. HTML is a markup language designed for the creation of web pages and other information viewable in a browser. HTML is based on an international standard (ISO/IEC 15445:2000) and the specification is maintained by the W3C.

To obtain the actual data, and not only maps, two more specifications were adopted: the WebFeatureService, which gives access to the actual data of geographic features, and the WebCoverageService, which makes space-varying phenomena available through coverages (which are values or properties of a set of geographic locations).

3.3 WebFeatureService Specification

A WFS publishes feature-level geospatial data to the web. This means that instead of returning an image, like a WMS does, the client directly obtains information about specific geospatial features of the underlying data, at both the geometric and the attributive levels. This interface uses XML over HTTP as delivery mechanism. More precisely, the Geography Markup Language (GML), a subset of XML, is used to express geographical features.

GML is another OpenGIS® specification from OGC. The latest version is 3.0[9] from January 29. 2003. It can serve as a modeling language for geographic systems as well as an open interchange format for geographic data, including both, the geometry and the properties of geographic features. There is a wide variety of object types for describing geography including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values.

The latest version of the WFS specification is 1.1, however a previous version, 1.0.0 (OGC IS, WFS 1.0.0, 2002)[10] from September 19, 2002, is used to describe the service because of the support of the software MapServer.

[8] URL to specification: http://www.w3.org/TR/html4/
[9] URL: http://www.opengeospatial.org/docs/02-023r4.pdf
[10] URL: http://www.opengeospatial.org/docs/02-058.pdf

This service is built analogous to the WMS. The WFS also supports the operation "GetCapabilities" to deliver metadata about the service. A basic WFS has to implement following operations:

- GetCapabilities - to describe its capabilities, especially which feature types it can serve and what operations are supported on them.
- DescribeFeatureType - to describe the structure of any feature type it can serve.
- GetFeature - to service a request to retrieve feature instances, encoded in GML.

A WFS could implement some more operations to become a transaction WFS but MapServer acts only as a basic WFS. The operations include data manipulation like create, update, and delete a feature. A basic WFS is also called a read-only WFS because data can only be read but not manipulated. In general, a WFS allows a client to combine data from multiple servers similar to the WMS.

The WFS is currently not of much interest for the KOMPSAT European Regional Archive because a satellite image is a raster-file and has no special feature instances. For the archive the WCS is of greater importance because users are enabled to retrieve the actual data which is acquired by the satellite.

3.4 WebCoverageService Specification

Unlike WFS, which returns discrete geospatial features, the WCS returns representation of space-varying phenomena like satellite images - i.e. it returns data with its original semantics (instead of pictures like the WMS) as coverages together with a detailed description. Coverages can be interpreted, extrapolated, etc. and not just portrayed like maps. At the moment, the software MapServer supports the current version 1.0.0 (OGC IS, WCS 1.0.0, 2003)[11] from August 27, 2003.

The WCS provides three operations:

- GetCapabilities - returns a XML document describing the service and a brief description of the data collections of which clients may request coverages.
- DescribeCoverage - returns a XML document containing a full description of one or more coverages served by a particular WCS.

[11] URL: http://www.opengeospatial.org/docs/03-065r6.pdf

- GetCoverage - generally run after the two operations above, it shows which requests are allowed and which data is available. This operation returns a coverage in a well-known format. The syntax and semantic is similar to the WMS GetMap and to the WFS GetFeature requests.

In this version the WCS is limited to describing and requesting grid, simple coverages with homogeneous range sets. Grid coverages are made of regularly spaced locations among the axes of a spatial coordinate reference system. The term "homogeneous range set" means that at each location in the domain, either a single value (e.g. panchromatic values from one band) or a series of values all defined in the same way.

The architecture of these OpenGIS® services can be seen in Figure 3.1 which is equivalent to Figure 1 in (OGC IS, WMS 1.1.1, 2002).

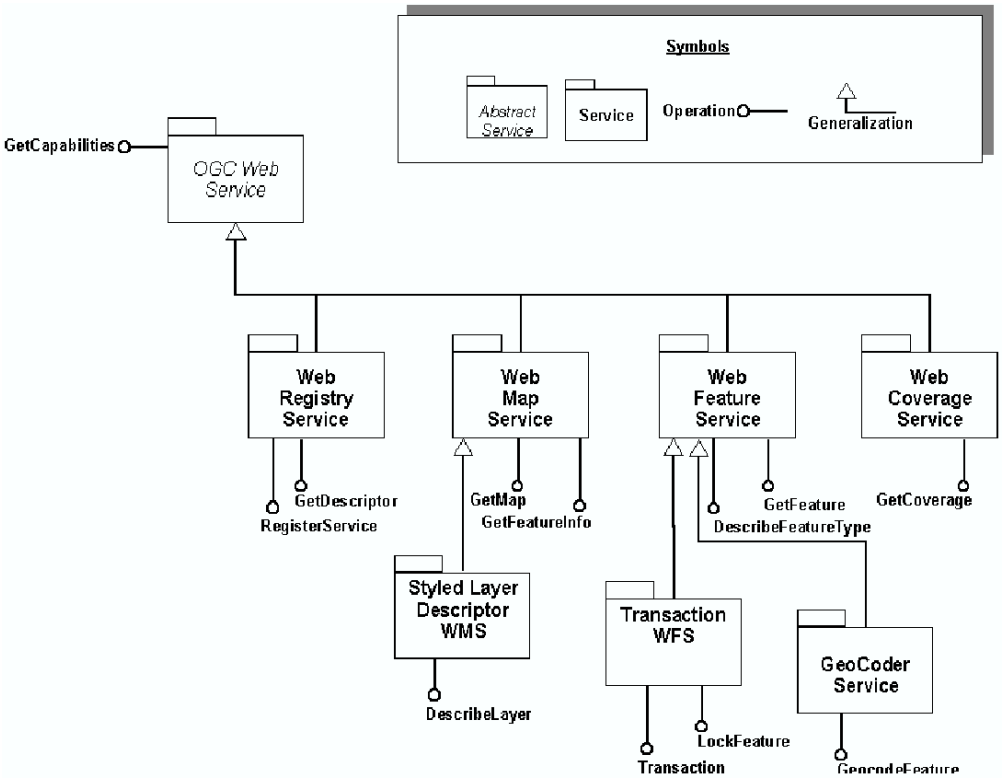


Figure 3.1: OGC Web Service Architecture diagram, from: (OGC IS, WMS 1.1.1, 2002), Figure 1

In order to use all these services and form valid requests some general rules have to be applied.

3.5 General HTTP Request Rules

Presently, the only platform explicitly supported by OGC Web Services (OWS) is the World Wide Web, more precisely Internet hosts implementing the HTTP protocol. Thus, all requests are HTTP URLs as defined by the W3C. HTTP supports the two request methods GET and POST but the basic WMS specification only defines GET for invoking operations. While parameters with HTTP GET are appended to the URL, parameters with HTTP POST are transported in the body of the request. POST requests usually result from a HTML form submission.

Inside an URL there are some reserved characters with special meaning. These characters are: "?", "&", "=", "/", ":", and ",". If they are not used as defined in table 3.1 they have to be escaped.

Char.	Reserved usage
?	Separator indicating start of query string.
&	Separator between parameters in query string.
=	Separator between name and value of parameter.
/	Separator between MIME type and subtype in format parameter value.
:	Separator between Namespace and Identifier in Spatial Reference System (SRS) parameter value.
,	Separator between individual values in list-oriented parameters.

Table 3.1: Reserved characters in HTTP GET URLs

A valid URL has to start with a prefix where protocol, hostname, path, etc. are specified, followed by parameters as name/value pairs. This means in detail:
`http://host[:port]/path?{name[=value]&}`
where `[]` denotes 0 or 1 occurrence of an optional part and `{}` any number of occurrences. The URL should be valid according to the HTTP Common Gateway Interface standard[12] (CGI).

Parameter names are not case sensitive, whereas parameters values are. The meaning is regardless of the order of the parameters (i.e. the request `http://myhost?SERVICE=WMS&VERSION=1.1.1` will result in the same response from the WMS as `http://myhost?VERSION=1.1.1&SERVICE=WMS`). Spaces in parameter values have to be escaped.

The OnlineResource-URL, which is specified in the capabilities document, is just a prefix. To generate a valid request, parameters are added to this prefix as described above.

All valid request parameters will be explained considering MapServer as

[12] URL: http://hoohoo.ncsa.uiuc.edu/cgi/

example. Thus, the software MapServer will be introduced in the subsequent section followed by all technical details for the parameters.

3.6 MapServer

To implement the OGC Web Services (OWS) described so far the software MapServer[13] is used at ARC-sr. MapServer was originally developed at the University of Minnesota (UMN), which is an associate member of the OGC, in the frame of a National Aeronautics and Space Administration (NASA) project for NASA. The choice was made to use MapServer because it is an OpenSource development and supports all required OWSs. The used version of MapServer is 4.6.0 which supports WMS 1.1.1 (client/server), non-transactional WFS 1.0.0 (client/server), and WCS 1.0.0 (server only).

Further significant advantages of this software are, the support of various vector formats e.g. Environmental Systems Research Institute, Inc. (ESRI) shapefiles and PostGIS, the support of many raster formats like GeoTIFF via the Geospatial Data Abstraction Library (GDAL), and the access to the C API through PHP/MapScript.

There are two basic ways to use this software, as CGI or through scripting language. The strategies at ARC-sr are either as server, then a CGI is invoked and performs all operations or as client, (see chapter 4), then PHP scripts provide access through PHP/MapScript.

3.6.1 Installation at ARC-sr

At ARC-sr the plan was to implement the services fully with OpenSource and free software. Thus, a Debian GNU/Linux 3.1 (sarge) operating system (OS) has been installed on the server http://spacey.arcs.ac.at. Debian uses the Linux kernel (the core of an operating system), but most of the basic OS tools derive from the GNU project, hence the name GNU/Linux. The installation includes the following packages and libraries: Apache HTTP server, PHP, GDAL, PROJ.4, GEOS, PostgreSQL and PostGIS.

The packages GDAL, PROJ.4, and GEOS have Debian packages and can thus be installed with Debian's package manager "apt". GDAL stands for Geospatial Data Abstraction Library and is a translator library for raster geospatial data formats e.g. GeoTIFF. PROJ.4 is a cartographic projections library used by e.g. PostGIS and MapServer for projection transformations. Most projections known by this software and their usage are explained in Evenden G. (2003). GEOS is an acronym for "Geometry Engine - Open Source". It is used by PostGIS for various functions regarding object manipulation and

[13] URL: http://mapserver.gis.umn.edu/

comparison. The package Apache is also a Debian package but some configuration settings have to be adjusted.

There are various other packages and libraries, e.g. libtiff, libpng, etc., but they are mostly part of the standard installation of Debian. If the package manager "apt" is installed properly the following commands can be used to make sure that all required packages are installed:

```
apt-get update
apt-get install libtiff4 proj proj-ps-doc curl lbcurl3-dev
  ibogre4 libgeos2 libgdal1 libgdal1-dev gdal-bin libgd-tools
  libgd2-xpm libreadline5-dev zlib1g-dev tcl8.4-dev libgeos-dev
  libxml2-dev libgd2-xpm-dev libwww0 libwww-dev libpng2 apache2
  make gcc
```

There are some software packages which need to be installed without using the package manager. These are: PDFlib, PostgreSQL, PostGIS, PHP, and of course MapServer. PDFlib, PostGIS and MapServer have to be installed by hand because there is no adequate package available in the official Debian sources. PostgreSQL was only available in a previous version (version number starting with 7) but the current version (starting with 8) is a lot faster and has several improvements, e.g. the usage of so called tablespaces (see section 2.4.1.2). PHP has to be installed as CGI and not, as it is default with Debian, as Apache module.

Furthermore, most of the installation commands, e.g. "`make install`", have to be executed with root (administrator) rights.

3.6.1.1 PDFlib

This library is used within MapServer to generate PDF output.

The used version of PDFlib is an outdated one (5) because of some restrictions in the current version (6). After downloading the source code[14] it can be compiled and installed with following commands:

```
tar -xvzf PDFlib-Lite-5.0.4p1-Unix-src.tar.gz
cd PDFlib-Lite-5.0.4p1-Unix-src
./configure
make
make install
```

[14] Available at: http://www.pdflib.com/products/pdflib/download/504src/PDFlib-Lite-5.0.4p1-Unix-src.tar.gz

3.6.1.2 PostgreSQL

PostgreSQL has already been introduced in section 2.2. For installation of the current version 8.0.1, which has no Debian package, following commands are required. Lines starting wiht "#" are comments and indented lines are the continuation of the previous line.

```
tar -xvjf postgresql-8.0.1.tar.bz2
cd postgresql-8.0.1
./configure --with-tcl
make
make install
# Install and remove the packages of the older version to create
# a user called postgres, to get the startscripts, etc.
apt-get install postgresql
apt-get remove postgresql
# Create symbolic links to all binaries
cd /usr/local/bin/
ln -s /usr/local/pgsql/bin/* .
# Create the system databases
su - postgres
cd /var/lib/postgres/data
rm -r *
initdb -D /var/lib/postgres/data/ --locale=C
# Use the configuration files from the package
rm pg_hba.conf
ln -sf /etc/postgresql/pg_hba.conf pg_hba.conf
rm pg_ident.conf
ln -sf /etc/postgresql/pg_ident.conf pg_ident.conf
mv postgresql.conf /etc/postgresql/postgresql.conf
ln -sf /etc/postgresql/postgresql.conf postgresql.conf
logout
vi /etc/postgresql/pg_hba.conf
# Add following lines to this file:
# local  all       wmsuser                                  trust
# host   wms_test  wmsuser 62.218.164.78  255.255.255.255  trust
# host   wms_test  wmsuser 172.24.11.255  255.255.255.255  trust
# In all other lines set method to md5
# Extract the Debian package
dpkg-deb -x /var/cache/apt/archives/postgresql_7.4.7-2_i386.deb
  /root/tmp/
cd /root/tmp/
cp etc/cron.d/postgresql /etc/cron.d/
vi /etc/cron.d/postgresql
# Change the only not commented line to:
# 2 0,5,10,15,20 * * 1-6 postgres if [ -z "`ps --no-headers -C
```

```
#   pg_autovacuum‘" -a -x /usr/local/pgsql/bin/do.maintenance ];
#   then /usr/local/pgsql/bin/do.maintenance -a; fi
cp usr/lib/postgresql/bin/do.maintenance /usr/local/pgsql/bin/
cp etc/logrotate.d/postgresql /etc/logrotate.d/
cp usr/lib/postgresql/bin/postgresql-startup /usr/local/pgsql/bin/
vi /usr/local/pgsql/bin/postgresql-startup
# Change some lines that the new ones look like:
# PGLIB=/usr/local/pgsql
# version=8.0
# export LANG=‘/usr/local/pgsql/bin/pg_controldata "$PGDATA" |
#   grep LC_CTYPE | cut -f 2 -d: | awk ’{print $1}’‘
# eval /usr/local/pgsql/bin/pg_ctl start -s -D ${PGDATA}
#   ${LOG_OPT} ${OPTIONS}
cp usr/share/postgresql/startup-checks-root.sh
  /usr/local/pgsql/bin/
vi /etc/init.d/postgresql
# Change one line to:
# $PREFIX/bin/startup-checks-root.sh
vi /usr/local/pgsql/bin/postgresql-startup
# Change socket directory to /tmp/ (from /var/lib/postgresql/)
vi /usr/local/pgsql/bin/do.maintenance
# Change socket directory to /tmp/ (from /var/lib/postgresql/)
```

Section 2.4.1.2 already explained how to create a database.

3.6.1.3 PostGIS

This extension to PostgreSQL is described in detail in sections 2.2 and 2.3.2. For installation a complete configured and built PostgreSQL source code tree is required. It is necessary to change to the directory which includes the PostgreSQL tree and to execute following commands.

```
cd postgresql-8.0.1/contrib
tar -xvzf postgis-1.0.0.tar.gz
cd postgis-1.0.0
make
make install
```

After unpacking the source code with “tar” but before compiling, the file “Makefile.config” needs to be edited. The changes shown in table 3.2 need to be made in order to compile the support for reprojection into PostGIS.

The description of how to add PostGIS support to an existing database can be read in section 2.4.1.2.

from	to
USE_PROJ ?= 0	USE_PROJ ?= 1
PROJ_DIR ?= /usr/local	PROJ_DIR ?= /usr
USE_GEOS ?= 0	USE_GEOS ?= 1
GEOS_DIR ?= /usr/local	GEOS_DIR ?= /usr

Table 3.2: Changes in file "Makefile.config" for PostGIS installation

3.6.1.4 PHP4

PHP needs to be installed as CGI and not, as it is default with Debian, as Apache module. There is the possibility to install the package's source code through "apt" and subsequently compile it. The following commands will install PHP in the appropriate way.

```
apt-get source php4
cd php4-4.3.10/
./configure --with-mysql --with-regex=system --with-gd
  --enable-force-cgi-redirect --with-zlib --with-dom
  --enable-dbase --with-pgsql --with-pdflib
make
make install
cp php.ini-dist /usr/local/lib/php.ini
vi /usr/local/lib/php.ini
# Change some variables:
# max_execution_time = 120
# max_input_time = 240
# memory_limit = 20M
# display_errors = Off
# log_errors = On
# error_log = /var/log/php
# session.save_path = /tmp/tmp_for_wms
```

3.6.1.5 MapServer

After unpacking some adaptations in the source code have to be made before going on with configuring and installing the MapServer software. In the file "mappostgis.c" the following changes need to be applied in order to use the layer type "tileindex" together with PostGIS layers. If there is a layer of the type "tileindex", trying to connect to a PostGIS database without these adaptations being applied, MapServer would produce an error ("Unsupported layer type in msPOSTGISLayerNext Shape()!"). Since the shapes of the tiled images are stored in the PostGIS database ARCS SATdata Catalog (see section 2.4.5) these changes are very important. Therefore it is necessary to change

the "switch"-statements in the functions "msPOSTGISLayerGetShape" and "msPOSTGISLayerGetShapeRandom" to (underlined lines are added):

```
switch(layer->type) {
    case MS_LAYER_POINT:
        result = force_to_points(wkb, shape);
        break;

    case MS_LAYER_LINE:
        result = force_to_lines(wkb, shape);
        break;

    case MS_LAYER_POLYGON:
        result = force_to_polygons(wkb, shape);
        break;

    case MS_LAYER_ANNOTATION:
    case MS_LAYER_QUERY:
        result = dont_force(wkb, shape);
        break;

    case MS_LAYER_RASTER:
        msDebug( "Ignoring MS_LAYER_RASTER in mappostgis.c\n" );
        break;

    case MS_LAYER_CIRCLE:
        msDebug( "Ignoring MS_LAYER_RASTER in mappostgis.c\n" );
        break;

    case MS_LAYER_TILEINDEX:
       result = force_to_polygons(wkb, shape);
       break;

    default:
       msDebug( "Unsupported layer type in msPOSTGISLayerNext
         Shape()!" );
       break;
}
```

The proper commands for the configuration and installation are as follows:

```
tar -xvzf mapserver-4.6.0.tar.gz
cd mapserver-4.6.0
./configure --with-proj=/usr/ --with-php=/usr/local/include/php/
  --with-gdal --with-ogr --with-wmsclient --with-wfs
  --with-wfsclient --with-pdf=/usr/local/include/
```

```
  --with-postgis=/usr/local/pgsql/bin/pg_config
  --disable-ignore-missing-data --enable-debug --with-wcs
  --with-geos=/usr/bin/geos-config --with-httpd=/usr/sbin/apache2
  --with-freetype=/usr/bin/freetype-config
make
vi /etc/ld.so.conf
# Add following line:
# /usr/local/lib
mkdir /usr/local/lib/php/extensions
cp mapscript/php3/php_mapscript.so /usr/local/lib/php/extensions/
vi /usr/local/lib/php.ini
# To load the MapScript extension make sure
# variable "extension_dir" is set to:
# "/usr/local/lib/php/extensions/" and add line:
# extension=php_mapscript.so
```

If more than 200 layers are required for one application the file "map.h" has to be edited and the variable "MS_MAXLAYERS" has to be changed to the number of layers wanted.

3.6.1.6 Apache

After the installation of all the required software the only tasks left to do are of configuration kind. First of all the HTTP server Apache has to be configured to handle PHP scripts, to handle CGIs requests, etc.

To allow the execution of CGI scripts the module "actions.load" needs to be activated. This is done by creating a symbolic link to the module in the directory "/etc/apache2/mods-enabled/".

```
ln -s /etc/apache2/mods-available/actions.load
  /etc/apache2/mods-enabled/actions.load
```

The main configuration file "/etc/apache2/apache2.conf" has to be edited for the use of Apache with PHP scripts. The following lines have to be added:

```
AddType application/x-httpd-php .php .phtml
ScriptAlias /php/ "/usr/local/bin/"
Action application/x-httpd-php /php/php
```

A line needs to be added to the file "/etc/apache2/sites-available/default" in order to associate a specific configuration file of MapServer (see section 3.6.2) to a specific CGI (application). "/cgi-bin/wms" is the CGI which has to be associated and "/home/wms_demo/demo.map" MapServer's configuration file.

```
SetEnvIf Request_URI "/cgi-bin/wms"
  MS_MAPFILE=/home/wms_demo/demo.map
```

See online documentation[15] for all further configuration issues of Apache.

3.6.1.7 Temporary Images

All instances of MapServer at ARC-sr store their temporary created images in the directory "/tmp/tmp_for_wms/". In order to delete the temporary files properly the file "/etc/cron.hourly/clean_tmps" needs to be created and made executable. Afterward the following lines have to be added. This script is then run hourly as a cron job.

```
#! /bin/sh
#
# Hourly cron job to clean temporary created images.
#
string1=‘date‘
string2=‘find /tmp/tmp_for_wms/ -name ’*.png’ -mmin +30 | wc -l‘
find /tmp/tmp_for_wms/ -name ’*.png’ -mmin +30 -exec rm -rf {} \;
string3=‘find /tmp/tmp_for_wms/ -name ’*.jpg’ -mmin +30 | wc -l‘
find /tmp/tmp_for_wms/ -name ’*.jpg’ -mmin +30 -exec rm -rf {} \;
string4=‘find /tmp/tmp_for_wms/ -name ’*.img.tmp’ -mmin +30 | wc
  -l‘
find /tmp/tmp_for_wms/ -name ’*.img.tmp’ -mmin +30 -exec rm -rf
  {} \;
echo "$string1: $string2 .png, $string3 .jpg and $string4
  .img.tmp files deleted." >> /var/log/tmpcleaner.log
```

3.6.2 The MapServer Configuration File (Mapfile)

The Mapfile is the heart of the MapServer. It defines the relationships between various objects, points MapServer to the data location, and defines how things are to be drawn.

The first object to mention is the "MAP" object, which defines the master object of the Mapfile, that is the one that holds all other objects (i.e. the "root"). It defines application- and map-wide parameters.

The main object is the "LAYER" object. The term layer, defined in the context of the Mapfile, means the combination of data plus styling. Data, in the form of attributes plus geometry, are combined with a style using "CLASS" and "STYLE" directives.

[15] URL: http://httpd.apache.org/

Other objects used in the template Mapfile, which can be found in detail in the appendix A.5, include: “OUTPUTFORMAT”, “SYMBOL”, “PROJECTION”, “LEGEND”, “LABEL”, “REFERENCE”, “SCALEBAR”, “METADATA”, and “WEB”.

This template Mapfile includes twelve layers. The first one prints a copyright statement, in the form of a visible watermark, in the top-left corner of every generated image. The second and third layer are drawing flags and outlines respectively of all available KOMPSAT-1 images. The next eight layers are divided into two types “TILEINDEX” and “RASTER” and are all for the KOMPSAT-1 image “eoc06182_20010216T091614”. The last layer has the function to apply some sort of “reuse-protection” or “copy-protection” to the image by painting a grid over images which are zoomed into a high scale (Figure 3.2). This is not a real “reuse-protection” or “copy-protection” since everybody who wants to copy the images can actually do so. But due to the quality degrade with the added grid lines the risk of misuse is minimized.

The image “eoc06182_20010216T091614” has, as previously explained, three overviews. Thus, four layers are required, one for every overview at a specific range of scale. The range of scale, where a layer is visible, is defined with the parameters “MAXSCALE” and “MINSCALE”. Because of the tiling there is the need for a second layer for every image which is of type “TILEINDEX” and does nothing more than fetching all required information from the database.

This connection to the PostGIS database and delivery of data is of major interest. The connection is established using the attributes “CONNECTION”, “CONNECTIONTYPE”, and “DATA” of the “LAYER” object. For the image “eoc06182_20010216T091614” at the first (best or highest) scale this looks like follows:

```
CONNECTION "user=wmsuser dbname=ksat host=/tmp"
CONNECTIONTYPE POSTGIS
DATA "the_geom from (SELECT a.oid as oid, a.the_geom as the_geom,
('/home/ksat/wms_data/tiles/K1/eoc06182_20010216t091614/' || a.lo
cation) AS location from eoc06182_20010216t091614 a WHERE a.locat
ion~''^eoc06182_20010216t091614_scale1_') as i USING UNIQUE oid US
INGSRID=4326"
```

The actual definition of the styling of the data is given in the layers with type “RASTER”. Only pixels with a value greater than zero are painted. These layers need a “PROJECTION” object and some information in the “METADATA” object. Since ARC-sr decided to store all images in simple longitude/latitude projection using WGS84, the definition in the Mapfile reads as `"init=epsg:4326"` (see section 2.3.1).

The data of the image stored in the “METADATA” object are the extent, the projection, and the title of the layer in a human readable form. For the image at the first (best or highest) scale there exists some additional data

because this layer should also be visible for the WCS. This additional data includes size, count of bands (one for the panchromatic KOMPSAT-1 images and five for the future KOMPSAT-2 ones), format, and explanations of the "rangeset" of the band.

More detailed information on using and creating a Mapfile is given in online documentation[16]. MapServer developers along with a team are currently working on a new version of the MapServer homepage where additional documentation[17] regarding the Mapfile can be found.

3.7 Request URL Parameters

This section describes the parameters and their usage for different request URLs. The GetCapabilities request of the OWS's WMS and WCS will be considered as example as well as WMS's GetMap, WCS's DescribeCoverage and GetCoverage request. All examples provided are based on the Mapfile attached in appendix A.5.

3.7.1 WMS GetCapabilities Request

The available parameters and their usage are explained in table 3.3.

Request Parameter	**Required/ Optional**	**Description**
VERSION=version	O	Request version. This optional parameter will be used in version negotiation between server and client.
SERVICE=WMS	R	Service type, must be WMS. Allows the same URL prefix to offer capabilities XML for multiple OWSs.
REQUEST=GetCapabilities	R	Name of request type.
UPDATESEQUENCE=string	O	Sequence number or string for cache control, capabilities version, (not implemented in MapServer).

Table 3.3: Parameters of WMS GetCapabilities request URL

To place a GetCapabilities request to the WMS at ARC-sr, the following

[16] URL: http://mapserver.gis.umn.edu/doc46/mapfile-reference.html
[17] URL: http://ms.gis.umn.edu/docs/reference/mapfile

URL can be used:

```
http://spacey.arcs.ac.at/cgi-bin/wms?SERVICE=WMS&VERSION=1.1.1&RE
QUEST=GetCapabilities
```

The returned capabilities XML of this request can be explored in the appendix A.6. After general service description (name, title, contact, etc.) capabilities of this service instance are described. First, different request types and the exception-format are explained, followed by all available layers. The definition of the layer which is explained in section 3.6.2 reads as follows:

```
<Layer queryable="0" opaque="0" cascaded="0">
  <Name>eoc06182_20010216t091614_scale1</Name>
  <Title>eoc06182_20010216T091614 Scalenr.: 1</Title>
  <SRS>EPSG:4326</SRS>
  <LatLonBoundingBox minx="16.0251" miny="47.2766" maxx="16.8179"
 maxy="48.8969" />
  <BoundingBox SRS="EPSG:4326" minx="16.0251" miny="47.2766" maxx
="16.8179" maxy="48.8969" />
  <ScaleHint min="0" max="18.6699" />
</Layer>
```

"Name" is used for machine-to-machine communication while "Title" is for the benefit of humans. This information is required to form a valid GetMap request.

3.7.2 WMS GetMap Request

All required URL parameters for this request are explained in table 3.4.

There are some additional and some changed parameters if the WMS is extended with Styled Layer Descriptor (SLD). SLD is an extension to allow user-defined symbolization of feature data instead of named Layers and Styles from the providers side. In brief, a SLD-enabled WMS retrieves features from a WFS and applies explicit styling information provided by the user in order to render a map.

Sending following URL to the WMS at ARC-sr is responded with the image seen in figure 3.2. As can easily be seen the name and the Spatial Reference System (SRS) of the requested layer are the same as in the capabilities XML.

```
http://spacey.arcs.ac.at/cgi-bin/wms?SERVICE=WMS&VERSION=1.1.1&RE
QUEST=GetMap&LAYERS=eoc06182_20010216t091614_scale1&STYLES=&SRS=E
PSG:4326&BBOX=16.35,48.19,16.39,48.23&WIDTH=375&HEIGHT=375&FORMAT
=image/png
```

Request Parameter	Required/ Optional	Description
VERSION=version	O	Request version.
SERVICE=WMS	R	Service type, must be WMS.
REQUEST=GetMap	R	Name of request type.
LAYERS=layer_list	R	Comma-separated list of one or more map layers.
STYLES=style_list	R	Comma-separated list of one rendering style per requested layer.
SRS=namespace:identifier	R	Spatial Reference System.
BBOX=minx,miny,maxx, maxy	R	Bounding box corners (lower left, upper right) in SRS units.
WIDTH=output_width	R	Width in pixels of map picture.
HEIGHT=output_height	R	Height in pixels of map picture.
FORMAT=output_format	R	Output format of map.
TRANSPARENT=TRUE or FALSE	O	Background transparency of map (default=FALSE).
BGCOLOR=color_value	O	Hexadecimal red-green-blue color value for the background color (default=0xFFFFFF).
EXCEPTIONS=exception_format	O	The format in which exceptions are to be reported by the WMS (default=SE_XML).

Table 3.4: Parameters of WMS GetMap request URL

3.7.3 WCS GetCapabilities Request

Since the WCS is of special interest for the work with satellite images it is introduced as a further OWS. The first request type is the GetCapabilities request which is almost the same as for WMS. The only difference is the additional parameter "SECTION" which allows the request of only a part of the capabilities document. All request parameters are explained in detail in table 3.5.

A valid request follows. As can easily be seen it is almost the same as for the WMS.

```
http://spacey.arcs.ac.at/cgi-bin/wms?SERVICE=WCS&VERSION=1.0.0&RE
QUEST=GetCapabilities
```

The most important part of the returned capabilities XML is the "ContentMetadata" section. Again, the same layer as previously used appears. The whole capabilities XML is shown in the appendix A.7.

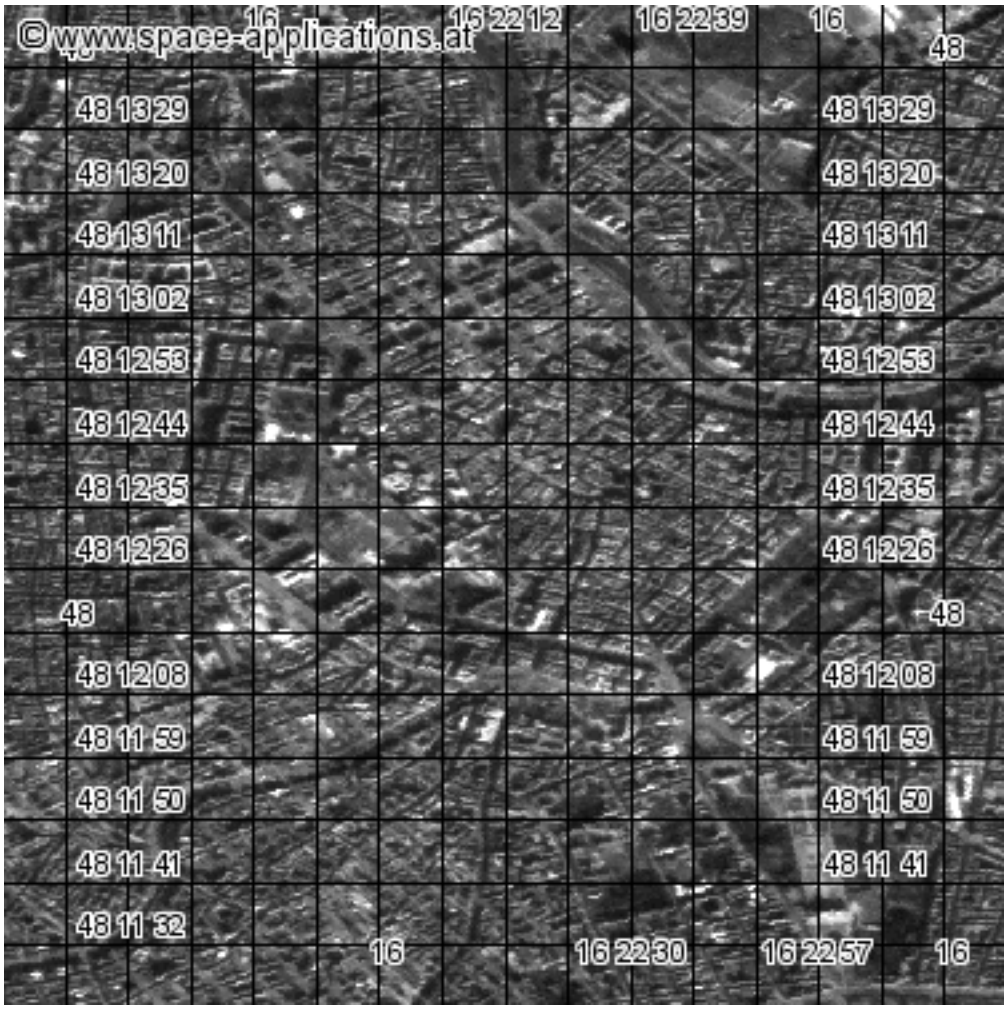


Figure 3.2: Image showing the city-center of Vienna, Austria received from ARC-sr WMS with the example "GetMap" request

```
<ContentMetadata>
  <CoverageOfferingBrief>
  <name>eoc06182_20010216t091614_scale1</name>
  <label>EOC 06182 February 16, 2001</label>
    <lonLatEnvelope srsName="WGS84(DD)">
      <gml:pos>16.0251 47.2766</gml:pos>
      <gml:pos>16.8179 48.8969</gml:pos>
    </lonLatEnvelope>
  </CoverageOfferingBrief>
</ContentMetadata>
```

Request Parameter	Required/ Optional	Description
REQUEST=GetCapabilities	R	Name of request type.
VERSION=1.0.0	O	Request version, 1.0.0 is the only available one at the moment.
SERVICE=WCS	R	Service type, must be WCS.
SECTION=/WCS_Capabilities/Service or /WCS_Capabilities/Capability or /WCS_Capabilities/ContentMetadata	O	Section of capabilities document to be returned.
UPDATESEQUENCE=string	O	Sequence number or string for cache control, capabilities version, (not implemented in MapServer).

Table 3.5: Parameters of WCS GetCapabilities request URL

3.7.4 WCS DescribeCoverage Request

The URL parameters are described in table 3.6. They are similar to the GetCapabilities request, apart from the addition of the opportunity to select some specific coverages for description.

Request Parameter	Required/ Optional	Description
REQUEST=DescribeCoverage	R	Name of request type.
VERSION=1.0.0	O	Request version.
SERVICE=WCS	R	Service type, must be WCS.
COVERAGE=coverage_list	O	A comma-separated list of coverages to describe. Default is all coverages.

Table 3.6: Parameters of WCS DescribeCoverage request URL

The subsequent request retrieves the coverage description for the image "eoc06182_20010216T091614" at scale one, which is the original scale.

```
http://spacey.arcs.ac.at/cgi-bin/wms?SERVICE=WCS&VERSION=1.0.0&RE
QUEST=DescribeCoverage&COVERAGE=eoc06182_20010216t091614_scale1
```

The whole XML can be explored in the appendix A.8. The most important line of the response is the following one.

```
<formats>GeoTIFF</formats>
```

This line states that coverages can be requested in GeoTIFF format, and therefore georeferenced. This is the significant difference to the WMS. With the WCS users are able to get the original raster data in GeoTIFF, georeferenced, format.

3.7.5 WCS GetCoverage Request

This request allows users to retrieve large raster data sets. The URL parameters are explained in table 3.7.

It is further of significance, that the so-called CRS is almost the same as the SRS in WMS's GetMap request, but it additionally allows the use of images where relationship to Earth coordinates may not be well defined, e.g. "raw-data" images. Another difference is the use of BBOX and TIME, and WIDTH/HEIGHT and RESX/RESY. While none, one or both of BBOX and TIME are allowed, WIDTH/HEIGHT and RESX/RESY are mutually exclusive. Therefore, just one of both is allowed but required.

A valid URL is:

```
http://spacey.arcs.ac.at/cgi-bin/wms?SERVICE=WCS&VERSION=1.0.0&RE
QUEST=GetCoverage&COVERAGE=eoc06182_20010216t091614_scale1&CRS=EP
SG:4326&BBOX=16.35,48.19,16.39,48.23&WIDTH=675&HEIGHT=675&FORMAT=
GeoTIFF
```

If the parameters WIDTH and HEIGHT are changed to:

```
&RESX=0.00005925261584454 43&RESY=0.0000594823788546256
```

the result will be almost the same, because 675 *pixels* nearly equal a difference of 0.04 in latitude or longitude. The value of 0.0000592526158445443 for RESX calculates as:

$$1\ px \mathrel{\widehat{=}} 6.6\ m \cong \frac{6.6}{2 * 6378137 * \pi/360}^{\circ} \approx 0.0000592888^{\circ}$$

The values do not match exactly because Earth is considered to be a spheroid rather than a sphere (see section 2.3.1). Therefore, one pixel in the image, which is 6.6 m on Earth ground due to KOMPSAT-1's spatial resolution, equals a change of 0.0000592888° in longitude at the equator. 6378137 m represents the equatorial semi-major axis of the used ellipse. Calculating the perimeter of Earth at the equator and setting it into relation with 6.6 m is shown by the previous formula.

Request Parameter	**Description**
REQUEST=GetCoverage	Name of request type. *Required.*
VERSION=1.0.0	Request version. *Optional.*
SERVICE=WCS	Service type, must be WCS. *Required.*
COVERAGE=coverage	Name of an available coverage. *Required.*
CRS=crs_identifier	Coordinate Reference System in which the request is expressed. *Required.*
RESPONSE_CRS=crs_identifier	Coordinate Reference System in which to express coverage responses. *Optional, defaults to the request CRS.*
BBOX=minx,miny,maxx, maxy,minz,maxz	Request a subset defined by the specified bounding box, with min/max coordinate pairs ordered according to the CRS parameter. *One of BBOX or TIME is required.*
TIME=time1,time2,... or TIME=min/max/res,...	Request a subset corresponding to the specified time instant or interval, expressed in an extended ISO 8601 syntax. Optional if a default time (or fixed time, or no time) is defined for the selected layer. *One of BBOX or TIME is required.*
PARAMETER=val1,val2,... or PARAMETER=min/max/res	Included only for range sets with compound values and thus uninteresting for KOMSAT-1 images. *Optional.*
WIDTH=w (integer) HEIGHT=h (integer) [DEPTH=d (integer)]	Request a grid of the specified width (w), height (h), and [for 3D grids] depth (d) (integer number of gridpoints). *Either these or RESX, RESY, [for 3D grids] RESZ are required.*
RESX=x (double) RESY=y (double) [RESZ=z (double)]	[when requesting georectified grid coverages] Request a coverage subset with a specific spatial resolution along each axis of the reply CRS. The values are given in the units appropriate to each axis of the CRS. *Either these or WIDTH, HEIGHT, and [for 3D grids] DEPTH are required.*
FORMAT=output_format	Requested output format of Coverage. Must be one of those listed under the description of the selected coverage. *Required.*
EXCEPTIONS=exception_format	The format in which exceptions are to be reported by the WCS (default=SE_XML). *Optional.*

Table 3.7: Parameters of WCS GetCoverage request URL

The correct values of the parameters RESX/RESY, for obtaining an image in original resolution, can be calculated from the values returned in the DescribeCoverage XML(see appendix A.8). The lines in the Mapfile responsible for these values are the entries “extent”, “srs”, and “size” in the METADATA object. The following lines provide an example.

```
"ows_extent"          "16.0251 47.2766 16.8179 48.8969"
"ows_srs"             "EPSG:4326"
"wcs_size"            "13380 27240"
```

If the parameters BBOX and TIME are omitted and RESX/RESY is used, the whole image is retrieved as one big GeoTIFF image. If the previously used image “eoc06182_20010216T091614” is requested in the original resolution, the resulting georeferenced GeoTIFF image needs approximately 350 MB of storage place.

A further advantage is the possibility to select individual bands and additionally to select a range within one band. This is not requested for KOMPSAT-1 but will possibly be of interest for KOMPSAT-2 because of its five bands.

3.8 Interaction between MapServer and PostGIS

As previously mentioned (section 3.6.2) the connection from the MapServer software to the PostGIS database is established through the attributes “CONNECTION”, “CONNECTIONTYPE”, and “DATA” of the “LAYER” object in the Mapfile. This is just one of the important connections between MapServer and PostGIS. Another important aspect is that all required data for the KOMPSAT layers is stored in the database. Thus, there is the possibility to generate a Mapfile on the fly out of the database.

A PHP script was written at ARC-sr to perform the creation of a Mapfile on the fly out of the database. This script is presented in the appendix A.9. The required parameters of the script are: “database”, “user”, “mapfile_templ”, and “mapfile_output”, whereby “database” is the database where all information is stored, “user” is a database username which has access to all required information, “mapfile_templ” is the name of a template Mapfile, and “mapfile_output” the output filename. The template Mapfile has to contain a valid Mapfile and may contain additional layers e.g. national boundaries.

The script uses MapServer’s PHP/MapScript to generate a valid Mapfile. First, an instance of the MAP object class is generated with the definitions found in the template file. Subsequently all new layers are generated and filled with information. Finally, the current state of the MAP object instance is saved to the output file. PHP/MapScript will be explained in more detail in section 4.4.

After the explanation of the OGC web service standards and the required software to implement them, it is time to show how to use them. Thus, the following chapter will cover the client side of these services by introducing the client developed at ARC-sr, followed by a chapter about service integration into different portals i.e. into the eoPortal and the SSE-Portal.

Chapter 4

ARC-sr Client

The previous chapter covered the server side of the OGC web services implemented at ARC-sr. The importance of the client side of these services is quite similar. Thus, this chapter will introduce a WMS client implemented at ARC-sr. Subsequently a chapter will analyze the service integration into different portals i.e. into the eoPortal and the SSE-Portal.

The ARC-sr client was implemented to show and extend the possibilities of the WMS. It is important that everybody is able to use the client with a standard web browser without having to install any software (i.e. plug-ins, applets, etc.). Therefore a mixture of JavaScript and PHP with PHP/MapScript scripts was developed and implemented. The JavaScript, which is executed only on the client, is required to get the intended functionality without enormous traffic between client and server.

The client can be accessed via any JavaScript enabled standard web browser via the URL: http://spacey.arcs.ac.at/KERA. Figure 4.1 shows a screenshot after entering the client. The user has to ensure, that the JavaScript functionality of the web browser is enabled and functions are allowed to be executed. It is also necessary to use a web browser that allows to view frames. The screen is divided into three parts or frames. The top part shows the title "KOMPSAT European Regional Archive" and various logos with links back as well as to related web pages. The remaining space on the left is covered by different control menus and on the right by the map and control tools as buttons.

4.1 Control Menus

The control menus include one for changing the current view, i.e. to adjust the size of the map, to set a zoom-factor, or to show the current position of the mouse on the map. This specific menu also shows a reference map. The red rectangle or cross (depending on the zoom level) shows the currently viewed

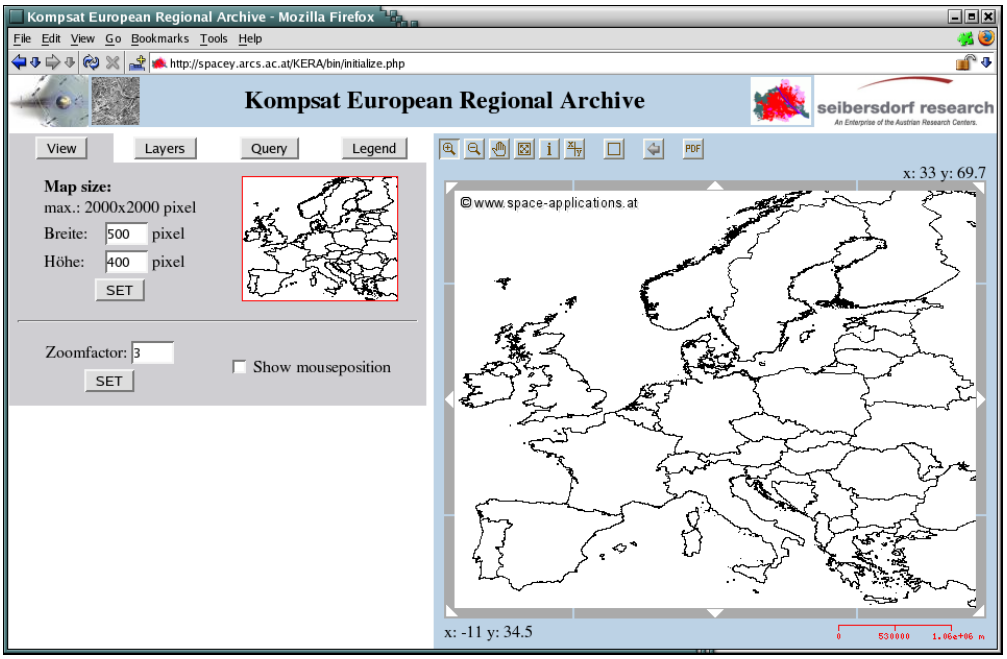


Figure 4.1: Screenshot of WMS client at ARC-sr - after startup

part and helps the user not to get lost in the map during zoom operations. Figure 4.1 shows this menu.

The second menu allows to manage the currently shown layers. On top there are various select boxes where layers can be selected for viewing. With the button "Add layers –>" they are added to the list of currently viewed layers right beneath. The other buttons can be used to adjust the layers' drawing order or to remove a layer. To show the current selection the button labeled "Redraw map" needs to be chosen. At the bottom there is another very interesting feature. Hereby, the user can provide the URL to another WMS instance and include layers from there. This triggers the WMS at ARC-sr to act as a Cascading Map Server. This menu can be seen on figure 4.2. A user can choose from preselected WMSes or provide a new URL.

The following menu, called query, enables the user to select a layer for a feature query. The second part of this menu is a so called "City Gazetteer". After entering the name of the required city a page presenting all cities, that are stored in the ARC-sr database and contain the searched string, is shown. Additionally to the names of the cities, geographic location in latitude and longitude is shown. This helps to choose the right city. Subsequently, the user can pan to the selected place and the resulting map is re-centered and highlights the name.

Besides, the values of an Area of Interest (AOI) used to query KERA can be adjusted. They can be entered manually however it is much easier to use the mouse, as described with other control tools later on. The final part "Possibility

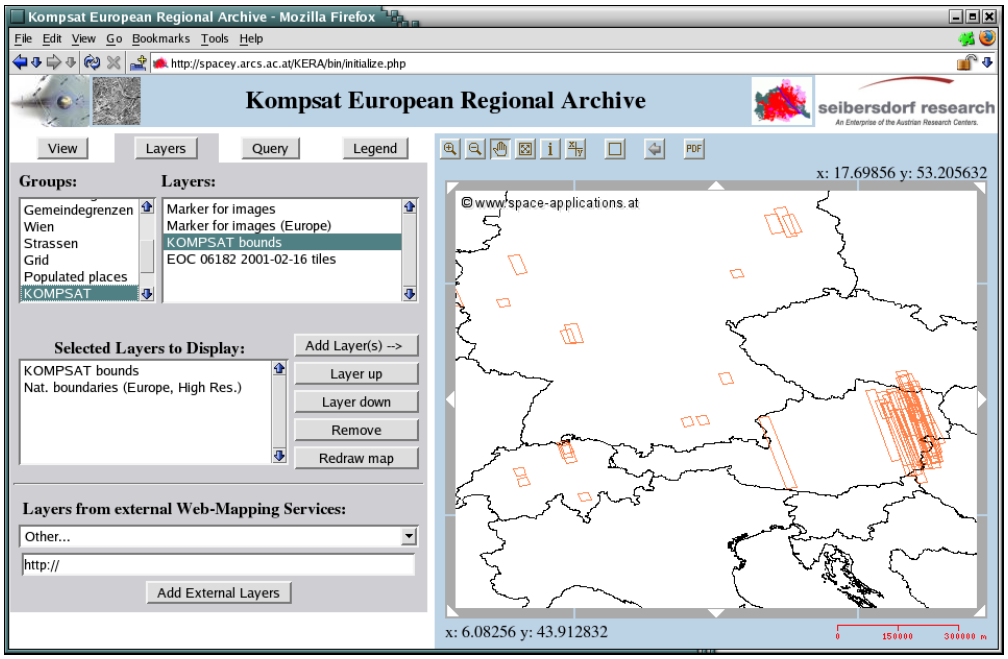


Figure 4.2: Screenshot of WMS client at ARC-sr - layer selection

to perform simple queries" is a fairly simple method to query the database and is only used for testing and debugging purposes. The provided input is taken as SQL which is executed in the database. Then the result is shown in a table. Figure 4.3 shows this menu with values for the AOI.

The last menu provides just a simple image showing a legend for the current map. This is helpful for the use of different vector layers or vector layers containing various features.

4.2 Map and Control Buttons

The right frame on the bottom shows the map, the control buttons, and information related to the map. The map allows interactive usage, e.g. zooming, panning, querying, etc. The functionality depends on the selected tool. At the top of this frame the user is able to select among several tools. The buttons are for zooming in, zooming out, panning or re-centering, zooming to original extent, drawing area for feature query, getting coordinate values of click, drawing AOI for query, going back one step, and getting current map as PDF. The currently selected button is highlighted as can be seen in figures 4.1 (zoom in tool), 4.2 (pan tool), and 4.3 (AOI tool).

If "zoom in" is selected the user has the possibility to zoom in to a point or to a rectangle. A click zooms in to the chosen point by the zoom-factor set in the menu. It is possible to create a red rectangle by clicking and holding

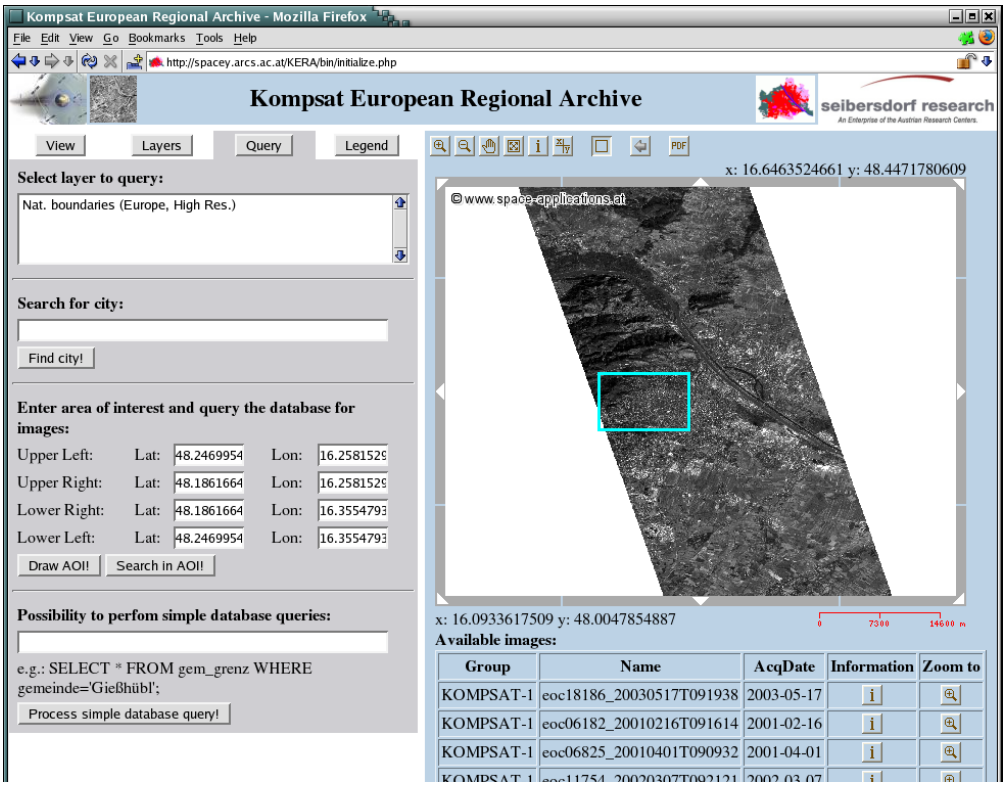


Figure 4.3: Screenshot of WMS client at ARC-sr - result of AOI query

the mouse button. Afterward clicking inside the rectangle will zoom the map in, while clicking outside will delete the rectangle. This is the same procedure as for drawing rectangular areas for a query. The pan tool allows the draging of the map. If released the map stops there and the requested information is loaded and displayed. Clicking with selected pan tool re-centers the map at the specific point.

The triangular arrows around the map are additional tools for panning. If one of the triangles is clicked, the map is moved the half of the current extent in the direction the triangle is showing. The numbers at the left/bottom and right/top corner of the map are showing the extent of the current map in longitude (x) and latitude (y). The red image at the right/bottom corner of the map shows a scalebar to determine distances on the map.

4.3 AOI Query

To perform a query for available KOMPSAT images, the user needs the tool for drawing an AOI. After drawing a rectangle and clicking inside, the database at ARC-sr is queried for available images (a click on the map will query a buffer area around the click). The result of such a query can be seen

in figure 4.3, where the queried area is outlined in turquoise. The coordinates could also be entered in the query menu in the left frame but this procedure is more complicated than drawing an AOI with the mouse. However, if the user needs precise coordinates, not achievable with the mouse, he can adjust the values in the query menu. With the button labeled "Draw AOI!" the AOI is just drawn and the user can correct errors. With the "Search in AOI!" the AOI is drawn but concurrently the search in the database is triggered.

Together with the result of a query the user gets the possibility of showing additional information (metadata) in a new window and to zoom to the extents of a specific image. Figure 4.4 shows the details for the previously used image "eoc06182_20010216T091614" provided by the client. Besides ordinary information like name, acquisition date, acquisition time, orbit, spatial extent, tilt angle, scene center etc. a browse image (quicklook) is provided. A button to close the page (not shown by the screenshot in figure 4.4) is provided at the bottom.

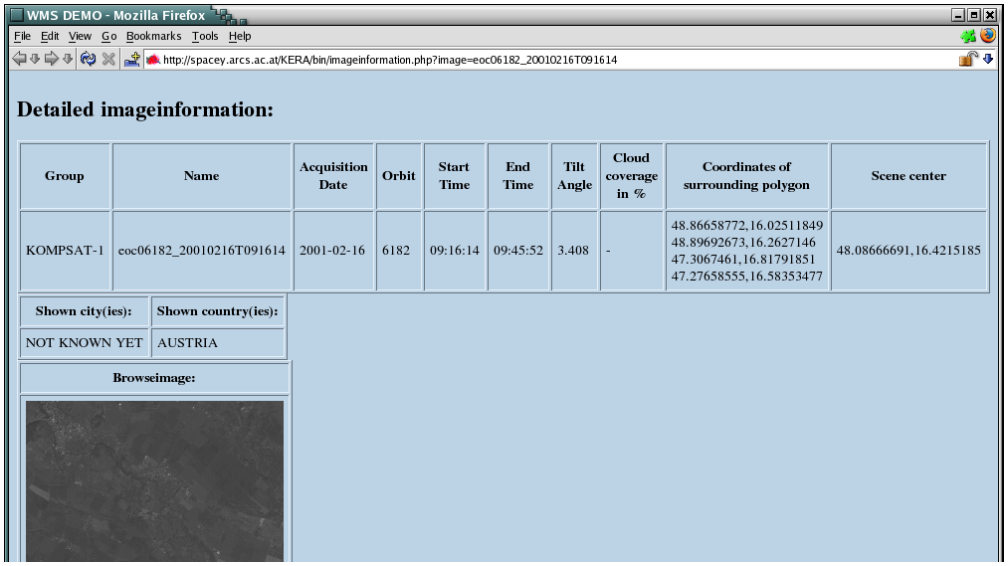


Figure 4.4: Screenshot of WMS client at ARC-sr - detailed image information

On the server side all images and HTML pages are generated by PHP scripts. On the client side the images and HTML pages just need to be viewed, however for interactive operations some JavaScript needs to be executed. The next two sections will explain these scripts in detail.

4.4 PHP with PHP/MapScript

PHP is the recursive acronym for "PHP: Hypertext Preprocessor". It is a commonly-used OpenSource scripting language that is especially made for web development.

Explaining all PHP scripts that are used by the client, is far beyond the scope of this thesis but some interesting results will be discussed subsequently.

The first interesting aspect is the integration of MapServer in PHP. If the PHP/MapScript extension of PHP is loaded and activated (see installation description in section 3.6.1.5) every PHP script can easily access all MapScript object classes, their members, and their methods (also called functions). First of all an instance of the MAP object class is required (see analogy to Mapfile in section 3.6.2). In order to create such an instance the constructor shown in the following line is required.

```
$map = ms_newMapObj($map_file);
```

`$map_file` has to be a PHP variable containing a string pointing to a valid and readable Mapfile. Otherwise an error message will be generated and the execution of the script stops.

Now, all members and methods of the MAP object class and some constructors, e.g. the LAYER object constructor, can be accessed.

Members are parameters or objects that are associated with a specific object and represent the data associated with them. They can in any case be read and sometimes even be changed. EXTENT is a member of the MAP object and also an object with some members. One of the members of the RECTANGLE object EXTENT is MAXX, which is the maximum longitude of the current MAP object. EXTENT is measured in longitude and latitude only if the PROJECTION object is set to "EPSG:4326". Since all examples are based on the Mapfile found in the appendix A.5 and the PROJECTION there is "EPSG:4326", MAXX is the maximum longitude. The following line shows how this value is retrieved in PHP:

```
$map->extent->maxx;
```

Methods, also called functions, can be used aswell. They access the data members of an object in predefined ways. The methods `zoomrectangle()` and `zoompoint()` are used to perform a zoom operation in the MAP object as shown below:

```
$map->zoomrectangle($rectangle,$width,$height,$map->extent);
$map->zoompoint($zoomfactor,$point,$width,$height,$map->extent);
```

The final methods, explained here, are those creating and storing an image in the file system. This is done with the functions `draw()` and `saveWebImage()`. An example is given below.

```
$image = $map->draw();
$image_url = $image->saveWebImage("MS_PNG",1,1,0);
```

If no error occurs, the PHP variable `$image_url` contains a string which is an URL pointing to the created image. This URL can be used in the HTML tag "IMG" as source attribute.

A script, which uses PHP/MapScript, can be found in appendix A.10. All the above examples are from this script. This script is responsible for the generation of all images required by the WMS client developed at ARC-sr.

A detailed description of the PHP/MapScript module, where all object classes, their members, and their methods are explained, can be found online (http://mapserver.gis.umn.edu/doc46/phpmapscript-class-guide.html).

A further interesting aspect is the use of PHP's PostgreSQL functions. With these the implementation of a gazetteer is accomplished at ARC-sr. The usage of this gazetteer is explained in section 4.1. The required functions are used by the example below:

```
$db_handle = pg_connect("user=".$db_user." dbname=".$db_name);
$query_result = pg_query($db_handle,"SELECT cityname,
X(the_geom), Y(the_geom) FROM cities WHERE
cityname~*'".$cityname."' ORDER BY cityname;");
while ($array = pg_fetch_row($query_result)) {
  echo "Cityname: ".$array[0].", Longitude: ".$array[1].
    "Latitude: ".$array[2];
}
pg_free_result($query_result);
pg_close($db_handle);
```

The function `pg_connect()` connects to a PostgreSQL database and returns a connection resource which can be used for further database manipulation. Such manipulation is done by the function `pg_query()` which executes a query and returns a result resource that can be accessed e.g. with `pg_fetch_row()`. This function returns the next row of the supplied query result resource as an array. Subsequently, the values in this array can be used by the script e.g. for display. If no longer required, the resources should be closed with the functions `pg_free_result()` and `pg_close()` to free used memory.

The whole functionality of PHP can be found in the online documentation http://www.php.net/manual/en/.

4.5 JavaScript

JavaScript is used for functionality and processes that need to be executed on the client. These include most of the interactive usage e.g. drawing the rectangle for the zooming, triggering the creation of new images, etc.

Programming with JavaScript is exhausting and time-consuming because there are various Document Object Models (DOM) used for different web browser. Even the same web browser with different version number implements different DOMs which leads to interoperability problems. DOM is an application programming interface to access HTML and XML documents. The need for a standard specification was seen by the W3C and a series of specification have been provided. The situation is getting better today with these standards in place but if the programmer also intends older web browser to be able to access a web page, he has to take care of several DOMs.

JavaScript is able to change most attributes of the objects stored in a document. Such objects are elements of a form, areas which are marked with the HTML tag "DIV", style elements, etc. To change e.g. the size of a "DIV" element the following lines are helpful:

```
var div = document.getElementById(elementname);
div.style.height = "200px";
```

These lines should be valid in all browsers, that implemented the W3C standards. The method `getElementById()` is a key method of the DOM-model from W3C. It returns the object whose id attribute is "elementname". The next line changes the height of this object to 200 pixels by using its style object. The style object allows access to stylesheet declarations. These declarations are made using Cascading Style Sheets (CSS), which is used to describe the presentation of a document written in a markup language like HTML.

The client interface is tested to work with Mozilla's Firefox, Microsoft's InternetExplorer and some versions of Netscape. However it generally works with all available web browsers.

The ARC-sr client interface is one possibility to access the KERA at ARC-sr. Other access points, that are available at the moment, are described in the following chapter.

Chapter 5

Service Integration

The WMS client at ARC-sr, as described in the previous chapter, is only one possibility to query the KOMPSAT European Regional Archive (KERA). Further possibilities exist via the eoPortal and via the SSE-Portal, both hosted by the European Space Agency (ESA). This chapter will introduce and describe these access points.

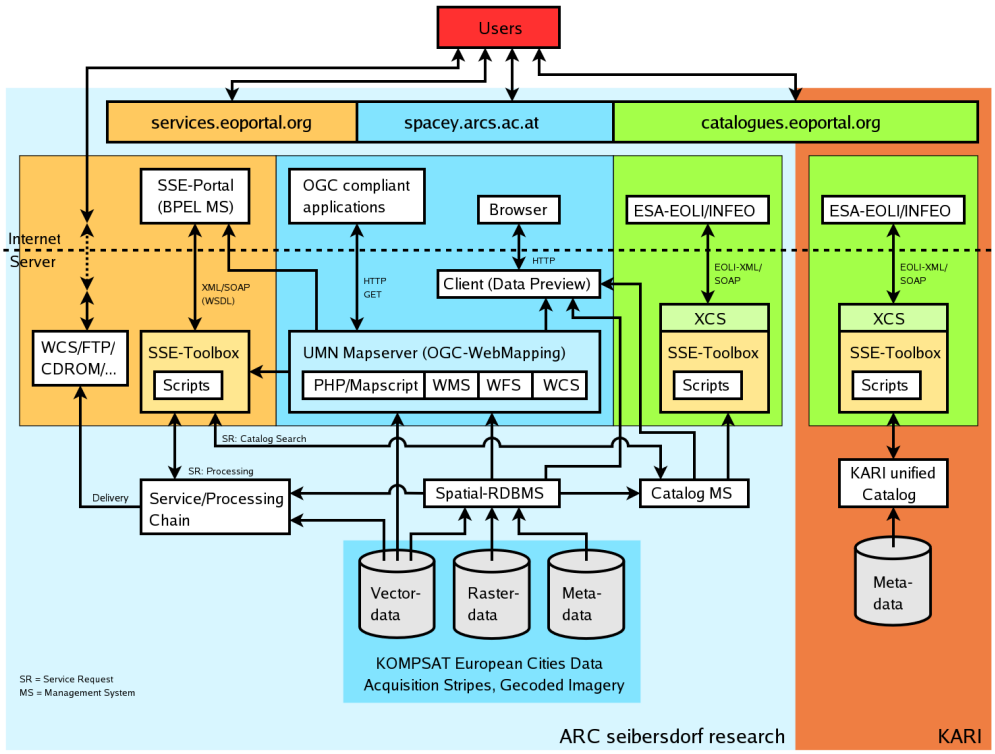


Figure 5.1: Interactions between the individual parts of the ARC-sr KERA implementation. Further the access of the unified catalog of the Korea Aerospace Research Institute (KARI) via the eoPortal.

Figure 5.1 shows the interaction between the individual parts of the implementation at ARC-sr including these new access points. The system can be divided into four functional units. The first three units with light-blue background represent ARC-sr whereas the dark-orange one Korea Aerospace Research Institute (KARI).

The first of the ARC-sr units (light-orange background) represents the access point via the SSE-Portal. The second one (blue background) includes all the WMS related components as well as the client discussed in the previous chapter. The third one (green background) shows all parts required for the access point via the eoPortal. Finally, the last unit on the right, actually a copy of the third one, represents the implementation at KARI which allows access to their international KOMPSAT archive also via the eoPortal.

At the basis is the spatially enabled ORDBMS situated, as introduced in chapter 2, serving the imagery and metadata. In the middle sits the software MapServer and all its components installed at ARC-sr, as discussed in chapter 3. Obviously, there are many connections and dependencies between these different parts and components. This strengthens the importance of standard specifications, as discussed in chapter 3.

5.1 SSE-Portal[1]

The first integration of KERA could be achieved into ESA's SSE-Portal. The objectives of the Service Support Environment (SSE) are:

- Facilitate access to Earth Observation (EO) data from ESA and other missions.
- Increase sustainability of EO data provision, by widening the range of offered data sources.
- Make operations of EO missions more efficient.

Using SSE, ESA provides a neutral service infrastructure, capable to foster the seamless integration of Earth Observation (EO) archives, products and services. Further it facilitates the set up of open operational services for EO in Europe. The SSE acts as service broker, which registers published service providers under particular service categories as well as search service provider, that allows users to search catalogs of connected data providers.

The SSE supports service providers by setting up their services. It enables the automatization of steps traditionally executed on the basis of manual interactions. Otherwise, they would unnecessarily complicate the service approaching an operational status and the same tasks would have to be repeated for tens

[1] URL: http://services.eoportal.org

or even hundreds of times. The SSE allows the integration of a wide range of heterogeneous EO and GIS services including catalogs of products. Significant advantages for service providers are: reduction in service set up and demonstration costs, possibility of combining and chaining service elements within the supply chain, easy re-use of generic or basic services, and improvement of existing services.

The main advantage for users is, that there exists a single portal for a wide range of services. Not only that catalogs are combined but complete service chain building blocks are made accessible via a common interface.

Furthermore, there exists a software called TOOLBOX. This software helps service providers to set up their services. SSE integrates service providers through XML, i.e. using SOAP and WSDL transported over the WWW. Thus, any service to be integrated within the SSE infrastructure has to expose a SOAP interface (described by a WSDL file) according to SSE ICD 1.4 (2005). The TOOLBOX helps the service provider to convert its service into a SOAP based service compliant to SSE specifications. Catalog services have to support EOLI-XML which is aligned to ISO/TC 211 (2003) and supported by TOOLBOX.

SOAP is a standard for exchanging XML-based messages over a computer network, normally using HTTP. SOAP forms the foundation layer of the web services stack, providing a basic messaging framework where more abstracted layers can be built on. WSDL is a XML format published for the description of web services.

Currently a service for searching the KERA catalog is implemented. The AOI to search for images is selected by using a client similar to the one described in chapter 4. The main difference is, that this portal is physically situated at ESA. Therefore, the search request is submitted over the WWW to ARC-sr using SOAP messages. There the TOOLBOX software awaits the request and invokes scripts depending on the request's type. A script is invoked for the search request from the SSE-Portal, executing a SQL query in the database and returning the results.

To use the search interface the user needs at least a JavaScript enabled web browser. There is also the possibility to use an advanced and extended version of the interfaces however, for this particular interface the user needs an installed Java plugin.

Figure 5.2 shows the search interface at the SSE-Portal. This interface has the OGC WMS specification implemented. Thus, it is possible to include the ARC-sr WMS in the map image as cascading map server. The orange image footprints and the images are served by the WMS located at ARC-sr. After defining an AOI (red rectangle), the search can be started by pressing the search button. Figure 5.2 shows the interface after a successful search. The images of the result, as shown below the map, can be selected for ordering. Clicking

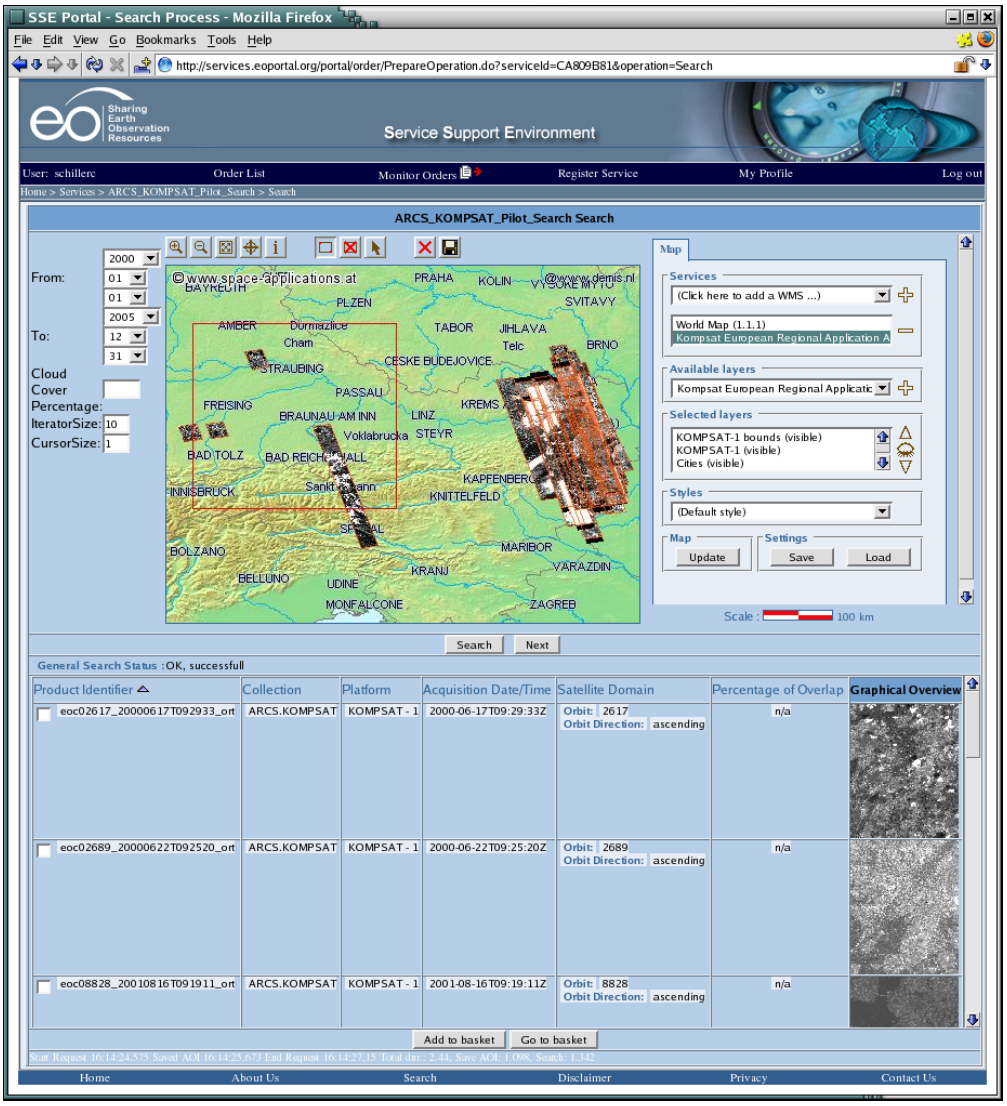


Figure 5.2: SSE-Portal: Search interface with search result presented in the lower section

on the quicklook image or the identifier name supplies a web page containing additional metadata information (Figure 5.3).

In addition to the search function for the catalog, an order service has been developed. With this service the user can order images and select a data delivery method. The available delivery methods are via FTP or via WCS (see section 3.4).

The script which is invoked through the TOOLBOX, connects to the KERA database, that follows the specification and design of the ARCS SATdata Catalog (see section 2.4), and performs a SQL query. An example query is shown in appendix A.11. The final lines include parameters entered by the user.

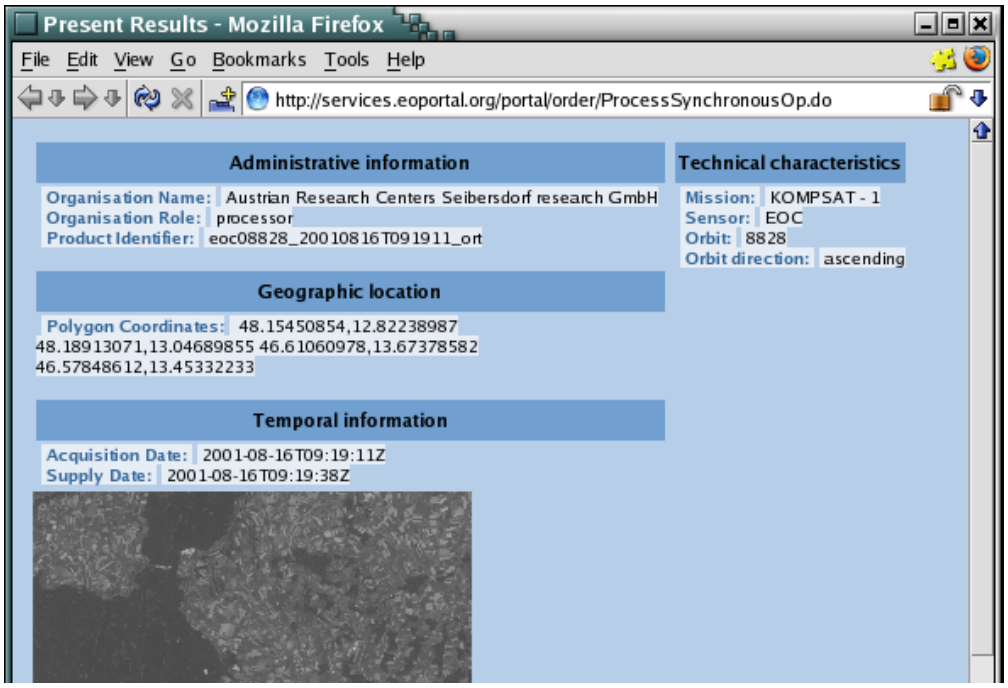


Figure 5.3: SSE-Portal: Additional metadata record of selected search result

5.2 eoPortal[2]

Furthermore, integration into the eoPortal could also be achieved. The eoPortal, also provided by ESA, is a non-brand portal providing an interface for space data providers to connect their catalogs. The KERA catalog at ARC-sr is now available as data collection and therefore searchable via the eoPortal catalog client.

The infrastructure used by the eoPortal catalog client is derived from the interoperable catalog system of the INFEO project. The Information about Earth Observation (INFEO) system, developed in the framework of the Centre for Earth Observation (CEO) program within the Joint Research Centre (JRC) of the European Commission, is now operated by ESA's European Space Research INstitute (ESRIN) in Frascati (Italy). It is based on a middleware implementation of a distributed search system and uses the Committee on Earth Observation Satellites (CEOS) "Catalog Interoperability Protocol" (CIP). Further, it provides gateways allowing access to IMS (i.e. the catalog protocol used by NASA) and GEO-profile based catalog servers. INFEO provides common access to EO catalogs and geographic datasets, and supports directory (i.e. collection), inventory and basic data item ordering services. INFEO offers the possibility to search metadata held by remote catalogs. Various catalogs can be searched simultaneously from a single user interface with a single set of input parameters.

[2]URL: http://catalogues.eoportal.org

The client is based on the EOLI (Earthnet Online) standard developed by ESA and the German Aerospace Center (DLR). This has proved to be a significant advantage, since SSE-Portal's TOOLBOX supports EOLI-XML for catalog queries. Now, there is the actual possibility of a convergence between INFEO and SSE catalog services based on EOLI-XML (Wernig-Pichler et al., 2005).

The eoPortal catalog client has approximately the same functionality as the SSE-Portal client and will run in any web browser having Java plugin, version greater than 1.4, installed. The main difference is that the eoPortal is originally intended for catalog searches on earth observation data products, while the SSE-Portal supports all kinds of web services. A further difference is that the eoPortal allows to search catalogs from numerous data providers at the same time. The data products of differing sources can be compared simultaneously. The client displays geographic coverage and quicklooks allowing the user to determine the data that best match the given requirements. In the SSE-Portal, however, just one service at the time can be invoked.

The eoPortal catalog client also provides two levels of searching. The first is the access to a database of data collections (via the discipline search). This allows the user to discover details of all data collections that are available for subsequent queries in order to identify individual data products of interest. If the user knows, which collection is needed, he can skip directly to the catalog search.

Figure 5.4 shows the client after successful catalog search. On the left side the available data collections are shown and can be selected for searches. The user is even able to choose more than one selection. On the map footprints of the returned images are shown, the green one is the currently selected one. On the bottom of the page there is a link (magnifier symbol) providing additional metadata information of the selected image (Figure 5.5).

Obviously KERA could be integrated into various systems namely WMS, SSE-Portal, and eoPortal. It reveals that these proven concepts might provide the basis for further developments. Some ideas for related future developments will be given in the following chapter.

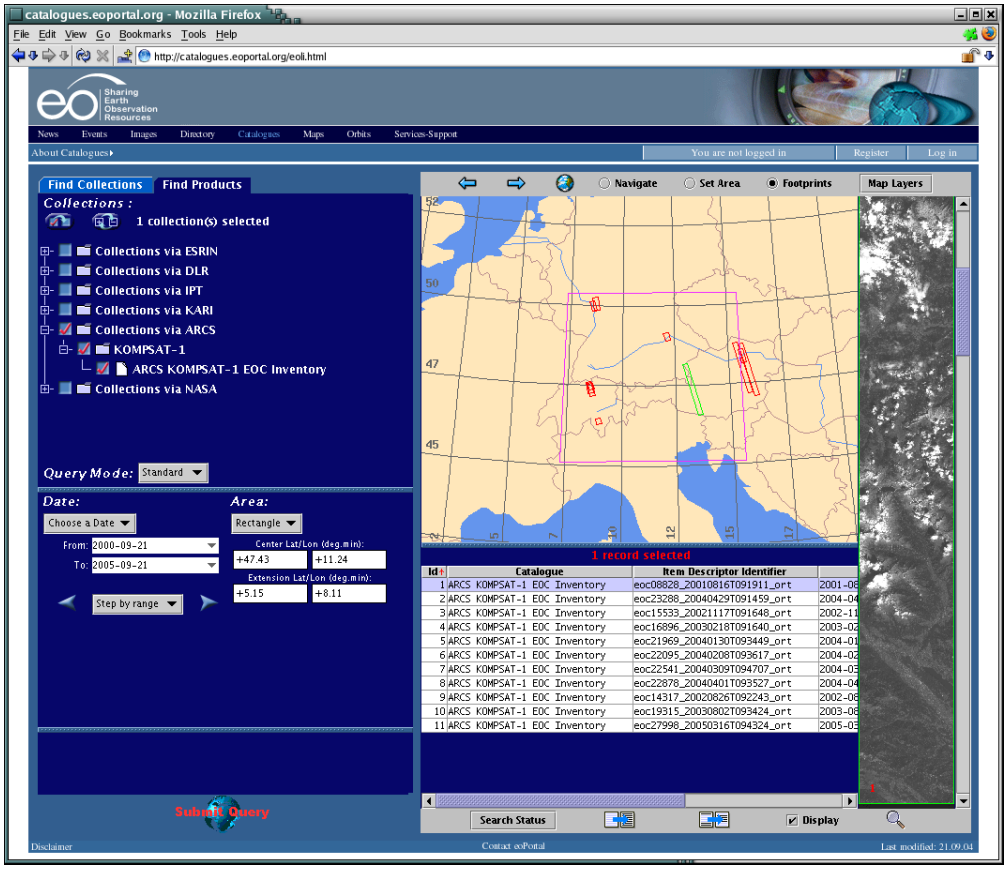


Figure 5.4: eoPortal: Search interface with search results (table) and quicklooks

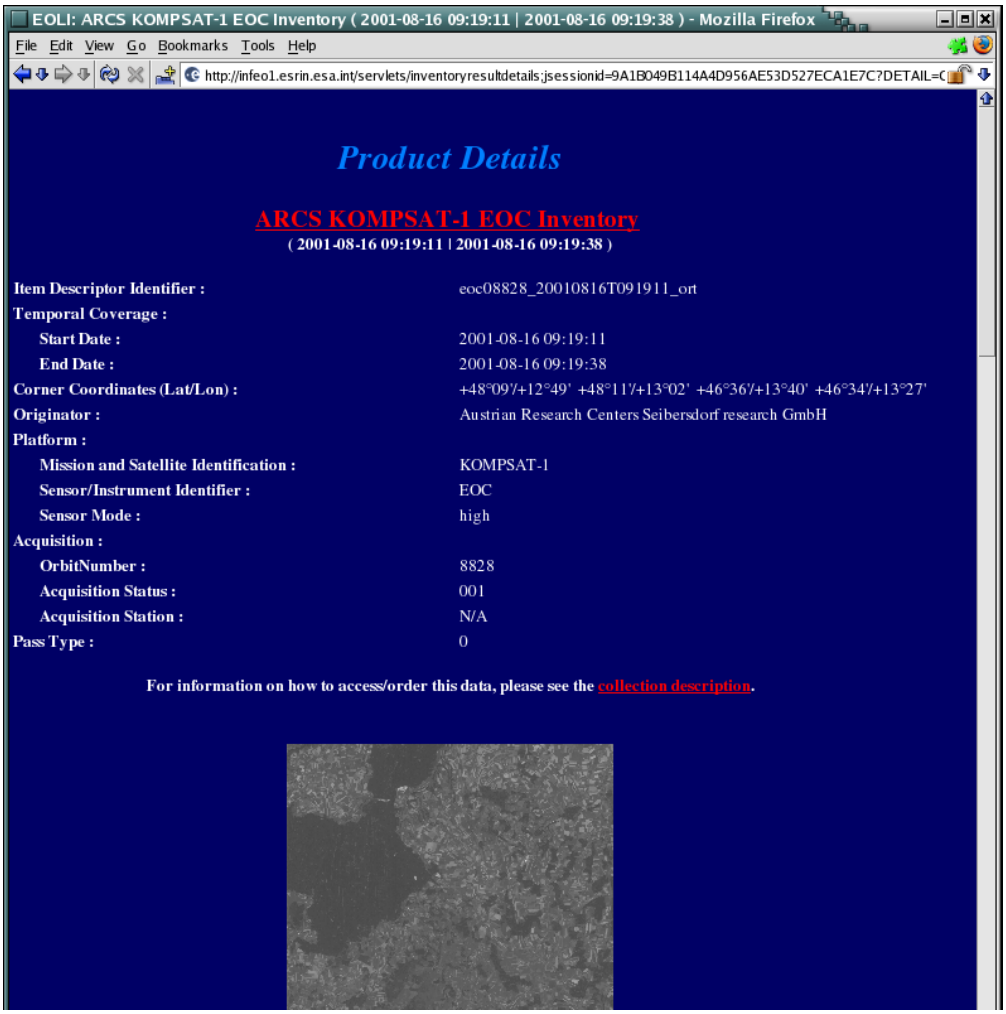


Figure 5.5: eoPortal: Additional metadata record of selected search result

Chapter 6

Conclusion and Scope

Where? This question was stated at the beginning of this thesis. Throughout this thesis it is discussed how Earth Observation (EO) data can be used answering this question. The Korean satellite KOMPSAT-1 and an outlook to the future satellite KOMPSAT-2 are provided as examples.

The first requirement is to develop and implement a database design suitable for EO data. This step is taken at ARC-sr with respect to existing standards and specifications. Problems occurring with geospatial data as well as software required to cope with them are explained. The installation of software packages, used for the implementation at ARC-sr, is introduced as well. Furthermore, a description of the steps following is made. Processes and workflow, when a new image is arriving, are explained. Performance and speed of the ARCS SATdata Catalog are improved in chapter 2.

Chapter 3 states the inevitable need for standardized and specified online web services. The OGC Web Services (OWS) WMS, WFS, and WCS are introduced and explained in detail. A general as well as detailed explanation of the installation at ARC-sr is illustrated with various examples.

Different possibilities of accessing EO data stored in KERA are described in chapters 4 and 5. First, the access point implemented and located at ARC-sr is introduced followed by access points via the SSE-Portal and via the eoPortal, both hosted by ESA.

Obviously, easy access to data is crucial for successful data dissemination. While accessing data has been significantly improved in the past few years, it is still cumbersome to find suitable data since numerous portals have to be searched and querying of archives is often limited. The use of satellite images has therefore been limited to expert users who own the know how and equipment to find, access, and process these data. This has additionally limited the acceptance of the use of satellite data in administration and other institutions. It is hoped that making satellite data more easy accessible and reducing the necessity to invest in software and time to learn the handling of it, increases

the acceptance and number of use cases and users significantly.

With the Internet being more and more part of our everyday lives, many barriers have already been removed and, assuming easy access, satellite images could be enjoyed by a greater audience. For example, in Europe the demand for easily accessible, online, very high resolution, geocoded information products is continuously increasing, driven by mass market applications which have cartographic data needs, e.g. car multimedia navigation systems and other navigation devices.

To integrate satellite derived data in these devices, standards, like those of OGC (explained in chapter 3), are required and used. There are a lot of different web services that will be integrated into mobile devices e.g. traffic news, electronic road atlas, timetables of public transport, availability of parking space in car parks, etc. There are several attempts of designing and implementing portals on the Internet (Lee et al., 2005) acting as service broker to clients. Service providers can register their services. This mechanism is called "publish-find-bind" and is illustrated in figure 6.1.

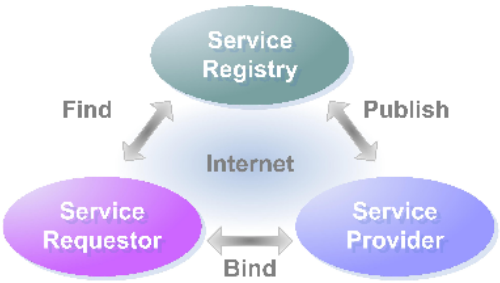


Figure 6.1: Illustration of the "publish-find-bind" mechanism of web services

There exist many different web services focusing on EO data. Chaining these existing services will create various new services and products. The SSE-Portal provides the ability to chain existing services. Due to the integration of many different web services focused on EO into the same portal with respect to existing standards, the data dissemination will, hopefully, grow and a greater number of users will use EO data. Sometimes they will not even realize to which services their devices connect but sometimes they will develop services by themselves that possibly integrate other services.

ARC-sr contributes to the evolution of standards for accessing the heterogeneous world of geoinformation. The standards are in constant development and ARC-sr contributes to feed the standards process with trials and verification activities. For this purpose ARC-sr builds application scenarios addressing the currently standards from different points of view.

As a first test case a product was designed and a pilot system implemented at ARC-sr, demonstrating the integration of various web services into one portal. This product is called Open Mobility and Geoinformation (OMGeo)-Portal

and is designed to provide destination information for travelers. The pilot implementation is available at http://spacey.arcs.ac.at/xeismobil. The integrated web services include services to obtain satellite images and aerial photographs acting as background for the map, to outline public transport routing and timetables to and from points of interest, to show car route information for points of interest, to give access to web-cams, to provide cultural and gastronomic information, etc. The pilot system is called GIS-Info Xeismobil and is a portal for a region in upper Styria, Austria (Gesäuse-Eisenwurzen). Figure 6.2 shows this pilot implementation.

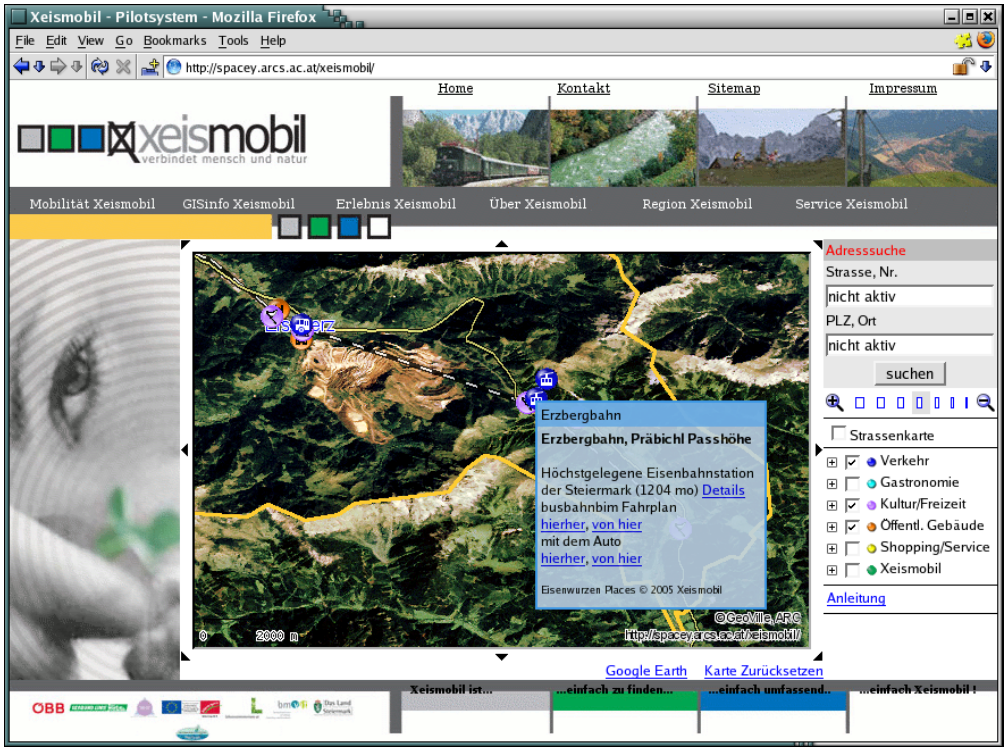


Figure 6.2: Screenshot of the pilot implementation of the OMGeo-Portal at ARC-sr

The greater data dissemination due to the higher interoperability and the more intense usage raises further requirements like Digital Rights Management (DRM). DRM can be defined as follows: DRM refers to the control and protection of digital intellectual property (content), including documents, images, video and audio. DRM limits what a user is able to do with that content even when he has possession of it.

The four main approaches of DRM are: physical copy protection, certificate-based encryption, product activation, and digital watermarking. Digital watermarking is technically not DRM, because it does not allow direct control over the use of the media. It does, however, at least prove the original ownership and provenance of the image.

While the W3C is developing an Open Digital Rights Language (ODRL) which is a proposed language for the DRM community, the OGC has installed a working group for Geospatial Digital Rights Management (GeoDRM).

This group states: "The lack of a Geospatial Digital Rights Management (GeoDRM) capability is a major barrier to broader adoption of web based geospatial technologies." The objectives for the GeoDRM working group are states as:

- Enable business models for web-based geospatial services by identifying or developing a trusted infrastructure for purchasing and protecting rights to digital content.
- Guide the development of OGC specifications and best practice recommendations to permit the exploitation of mainstream DRM approaches, technologies and standards wherever possible.
- Test, verify and mature the technologies required for geospatial DRM including electronic commerce and information security.
- Develop specifications for geospatial DRM that continue the OGC technical baseline.

There are no specifications available at the moment however work is in progress.

Digital watermarking is a technique that allows the owner to add hidden copyright notices or other verification messages to digital audio, video, or image signals and documents. This hidden message equals to a group of bits describing information pertaining to the signal or to the author of the signal (name, place, etc.). While the addition of the hidden message to the signal does not restrict the use of the signal, it provides a mechanism to track the signal back to the original owner.

Watermarks can be classified into two sub-types: visible and invisible. Often there is a visible watermark added in the shape of a copyright symbol ("©") (e.g. shown in figure 4.1). There are various technics and algorithms for adding invisible watermarks, for example by using spatial and frequency domain techniques. Solely spatial techniques are rather not robust to attacks on the signal, e.g. cropping and zooming, whereas most frequency domain techniques and mixed-domain techniques are much more robust.

Watermarks can be applied to most image formats available. There is also the new JPEG2000 standard, an image coding system that uses state-of-the-art compression techniques based on wavelet technology. Part 8 of the specification under development, called JPSEC (security aspects), wants to address topics like encryption, source authentication, data integrity, conditional access, ownership protection, etc. JPSEC will allow protection of content owner rights (copyright). This includes an ownership identification mechanisms robust

to malicious attacks and non-malicious processing of the JPEG2000 bitstream and/or the image it represents.

Further requirement raised, is the need for user management and accounting to restrict the access to the archive. DRM is required if the user already possesses a product in order to limit what he is able do with it. Therefore, user management and accounting is required to control who has access to the data, who gets possession of the data, and for the handling of payments, etc.

There are different models how EO data are sold. The simplest model is to pay a certain price per square kilometer after ordering data for a specific area. This data is new, when ordered, and shows everything as it is in reality. But what happens if changes occur in the area? The user would have to order the whole data again and again. This model is rather expensive but gives full possession of the data. Another model would be to pay just when data is really required. Hereby the user does not have to buy data for a whole country if he wants to drive through. Actually, he only needs data for the specific streets. Generally he does not know these streets from the start and therefore it would be the best if he orders the data when required, quasi on-demand. The model for the payment could be to pay per request or per amount of the transmitted data. The on-demand model can also be paid with flat-rate. This means paying a fixed price every month allowing unlimited or a certain amount of requests or data.

At ARC-sr the inevitable need for standard specifications was recognized. It has been discovered, that a lot of OpenSource software is available to fulfill this requirement. A concept how these software packages can be installed, used, and combined was designed, implemented and proven. This concept is explained and discussed throughout this thesis.

The aim of the development of the KOMPSAT European Regional Archive (KERA) and its different access points is to offer enhanced products based on the principles of a high degree of automation, easy access to data and services, and availability to both expert and non-expert users. I hope that by making satellite data more accessible and by reducing the necessity to invest in software and time to learn the handling of it, the acceptance and number of users and use cases can be greatly increased.

Glossary

Notation	Description
ANSI	American National Standards Institute 14
AOI	Area Of Interest, Area used for geospatial searches within the database. 64–67, 73
Apache	The Apache HTTP server is an OpenSource HTTP server for modern operating systems including UNIX and Windows NT. It is a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards. 45, 46, 49, 51, 52
API	Application Programming Interface, Set of definitions of the ways one part of computer software communicates with another. 8, 45, 92
apt	apt stands for Advanced Package Tool and is a management system for software packages used by Debian. . . 45, 46
ARC-sr	Austrian Research Centers - Seibersdorf research GmbH (Department of Intelligent Infrastructures and Space Applications) . iii, iv, xi, 1–3, 9–12, 14, 20–23, 31, 34, 35, 37, 39, 45, 52–55, 57, 61–67, 69–73, 75, 79–81, 83, 99, 100
attribute	In database management, an attribute is an inherent property in an entity or associated with that entity. 13
BoundingBox	A BoundingBox defines the portion of the Earth to be mapped, e.g. with a GetMap request to a WMS. 40
C	C is on of the most widely used programming languages. 8, 45, 92

Notation	Description
Cascading MS	A Cascading Map Server is a WMS that behaves like a client for other WMSes and like a WMS to other clients. A Cascading Map Server can combine the information of several different WMSes into one service. It can also be used for data conversion like changing the output format or transforming the coordinate reference system. . . 40, 64, 73
CEO	Centre for Earth Observation 75, 89
CEOS	Committee on Earth Observation Satellites 75
CGI	Common Gateway Interface, This is an important WWW technology that enables a client web browser to request data from a program executed on the web server. CGI specifies a standard for passing data between the client and the program. 44–46, 49, 51
CIP	Catalog Interoperability Protocol 75, 89
constructor	In Object-Oriented Programming (OOP) constructors are special instance methods that are called to create an instance of a class, an object. 68
coverage	Values or properties of a set of geographic locations. 41–43, 58, 60, 100
CPU	Central processing unit, Part of a computer that interprets and carries out the instructions contained in the software. 34
cron job	Cron is a daemon to execute jobs, several commands, on a fixed scheduled. 52
CRS	coordinate reference system, Coordinate system which is related to the real world by a datum. 16, 59, 60
CS	coordinate system, Set of mathematical rules for specifying how coordinates are to be assigned to points. 16
CSS	Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in a markup language. Its most common application is to style web pages written in HTML. The CSS specifications are maintained by the W3C. 70
date line	The international date line is mostly located on the line of longitude $\pm 180°$, on the side of the Earth that lies opposite the prime meridian. 18
datum	Parameter or set of parameters that define the position of the origin, the scale, and the orientation of a coordinate reference system. 16
DBMS	Database Management System 3, 13
Debian	Debian is a free OS for your computer. 45–47, 49, 85

Notation	Description
DLR	The German Aerospace Center (DLR) (German: Deutsches Zentrum für Luft- und Raumfahrt e.V.) is the national research center for aviation and space flight of the Federal Republic of Germany. 76, 87
DOM	The Document Object Model (DOM) is an application programming interface to access HTML and XML documents. It is programming language and platform independent. 70
DRM	Digital Rights Management 81–83
EER	Extended Entity-Relationship, An EER diagram is a diagram for high-level descriptions of logical data models, and it provides a graphical notation for representing such data models. 21
entity	Representation of a discrete object. Concerning relational databases, "entity" is often similarly used as "table". . 13
ENVI	ENVI is a software from RSI used for remote sensing applications e.g. data visualization, data analysis, etc. . . . 31
EO	Earth Observation . . . . . iii, 9, 72, 73, 75, 79, 80, 83, 89
EOC	Earth Observing Camera, A high-resolution images taking sensor on KOMPSAT-1. . . . . 1, 23
EOLI	Earthnet Online interface standard developed by ESA and DLR. 73, 76
eoPortal	Portal to Earth observation data from the ESA. iv, 9, 10, 13, 62, 63, 71, 72, 75–79
EPSG	European Petroleum Survey Group, Numbering system for geographic projection systems. 16
equator	Imaginary line drawn around the earth halfway between the poles. 15
ESA	European Space Agency . . . iv, 9, 12, 71–73, 75, 76, 79, 87
ESRI	Environmental Systems Research Institute, Inc. is one of the biggest firms which produce Geographic Information System software. One of their most famous products is called ArcView. 45
ESRIN	ESA's European Space Research INstitute (ESRIN) . . 75, 89
FAQ	Frequently Asked Questions 8
field view	Field view treats reality like a continuous two dimensional space. 12
Firefox	An OpenSource web browser, designed for standards compliance, performance and portability by the Mozilla project. 9, 27, 70

Notation	Description
flattening	Flattening of an ellipse f is defined as: $f = \frac{a-b}{a}$ where a denotes the equatorial semi-major axis and b the polar semi-minor axis. 16
FTP	The File Transfer Protocol (FTP) is a software standard for transferring computer files between machines with widely different operating systems. 74
gazetteer	A gazetteer is a geographic dictionary. It looks up coordinates by names of places or vice versa. 8, 64, 69
GCP	Ground Control Point, Points used to orthorectify satellite derived images. 3, 31
GDAL	GDAL stands for Geospatial Data Abstraction Library and is a translator library for raster geospatial data formats e.g. GeoTIFF. 45
generalization	Generalization, speaking of relational databases, means the inheritance of attributes from one superclass to some other classes. 23
GeoDRM	Geospatial Digital Rights Management 82
GEOS	Geometry Engine - OpenSource, Module used by PostGIS for a lot of functions regarding object manipulation and comparison. 17, 45
geospatial	Refers to some place on Earth's surface. Equivalent to geographic. 3, 12
GeoTIFF	Geographically Registered Tagged Image File Format . 3, 34, 45, 59, 61
GIF	Stands for Graphics Interchange Format and is a raster image format that is widely used on the WWW, both for still images and for animations. 41
GIS	Geographic Information System, Mostly defined as software system for managing spatial data and associated attributes. It is capable of integrating, storing, editing, analyzing, and displaying this data. 8, 37, 73
GML	Geography Markup Language, GML is a subset of XML used to express geographical features. It can serve as a modeling language for geographic systems as well as an open interchange format for geographic data, including both the geometry and properties of geographic features. 41, 42
GNU project	The GNU project was launched in 1984 to develop a complete UNIX like operating system which is free software. 45, 88, 89

Notation	Description
GNU/Linux	An UNIX-like operating system that is based on the Linux kernel combined with libraries and tools from the GNU project is called GNU/Linux. 45
GPL	The GNU General Public License is a free software license, originally written by Richard Stallman for the GNU project. It has since become the most popular license for free software. 99, 100, 122
grid coverage	Grid coverages are coverages which have a domain comprised of regularly spaced locations along the axis of a spatial coordinate reference system. 43
HR	high-resolution, Images with a spatial resolution between 3 and 10 *meter* are named high-resolution images. 1
HTML	HyperText Markup Language, This is a markup language designed for the creation of web pages and other information viewable in a browser. HTML is based on an international standard (ISO/IEC 15445:2000) and the specification is maintained by the W3C. . . 41, 44, 67, 69, 70, 86, 87
HTTP	Hypertext Transfer Protocol, Protocol used to convey information on the World Wide Web. The original purpose was to provide a way to publish and receive HTML pages. 39, 41, 44, 51, 73, 85, 94
IKONOS	IKONOS is a commercial earth observation satellite that collects very-high-resolution VHR imagery at the same resolution KOMPSAT-2 will do. 2, 7
INFEO	The Information about Earth Observation (INFEO) system, developed in the framework of the CEO program within the JRC of the European Commission, is now operated by ESRIN in Frascati (Italy). It is based on a middleware implementation of a distributed search system and uses CIP. INFEO provides common access to EO catalogs and geographic datasets. 75, 76
Internet	The Internet is the extensive, worldwide computer network available to the public. . . . . iii, 1, 89
interoperability	Geospatial interoperability is defined as the ability for two different software systems to interact with geospatial information mostly via the Internet. 38
ISO	International Organization for Standardization 12, 14, 21
ISO/TC	ISO Technical Committee 12

Notation	Description
ISO/TC 211	ISO Technical Committee responsible for Geographic information/Geomatics 12
Java	Java is a reflective, object-oriented programming language. Java should not be confused with JavaScript, which shares only the name. 73, 76
JavaScript	Scripting language best known for its use in websites. . 9, 27, 28, 63, 67, 69, 70, 73, 90
JPEG	JPEG is a commonly used standard method of compressing photographic images and stands for Joint Photographic Experts Group. 34, 41
JPEG2000	JPEG2000 is a new image standard, which is an image coding system that uses state-of-the-art compression techniques based on wavelet technology. 82, 83, 90
JPSEC	JPSEC is part 8 of the new JPEG2000 specification and stands for security aspects. This part of the specification is under development, and will address topics like encryption, source authentication, data integrity, conditional access, ownership protection, etc. 82
JRC	Joint Research Centre 75, 89
KARI	Korea Aerospace Research Institute . . . . . iii, 1, 71, 72, 90
KERA	KOMPSAT European Regional Archive . . . . . iii, 8, 9, 64, 70–76, 79, 83
kernel	The kernel is the core of an operating system. It is a piece of software responsible for providing secure access to the machine's hardware and to various computer processes. 45
KOMPSAT	Korean Multi Purpose Satellite operated by KARI. . 1, 2, 4, 7, 13, 16, 18, 19, 23, 42, 53, 54, 59, 61, 63, 66, 72, 79, 87, 89–91
latitude	Describes the location of a place on earth north or south of the equator. 15
layer	Representation of one set of data. 9, 40
logical schema	The logical or object schema is a data model of a specific problem domain which does not include the design considerations and physical storage parameters found in a physical schema. A logical schema contains entities made up of attributes, and connected by relations 13
longitude	Describes the location of a place on earth east or west of a north-south line called the prime meridian. 15

Notation	Description
map	Visualization of spatial data. 1, 93
Mapfile	The Mapfile is the heart of the MapServer. It defines the relationships between objects, points MapServer to where data are located and defines how things are to be drawn. 52–54, 61, 68, 99, 100
MapServer	Software originally developed at the University of Minnesota for constructing spatially enabled Internet applications. . . 8, 39, 41, 42, 44–46, 49, 51, 52, 54, 58, 61, 68, 72, 91, 93, 99, 100
markup language	A markup language combines text and extra information about the text. 39, 41
MB (Megabyte)	Unit to measure computer storage, equal to 1,024 kilobytes or 1,048,576 bytes. A byte is a sequence of 8 bits, which are the basic information units like one and zero, or true and false. 3
member	In Object-Oriented Programming (OOP), data members of an object represent the data associated with it. . 68, 69
meridian	Line of constant longitude. 20
metadata	Data about data. Stores what is stored, where it is stored, etc. . . . . iii, 3, 11, 12, 74
method	In Object-Oriented Programming (OOP), methods, sometimes also called functions, access the data members of an object in predefined ways. 68, 69
MIME type	It was originally defined for email, but is reused in other Internet protocols such as HTTP. A MIME type consists of the combination of type and subtype separated by a slash, e.g. text/plain, text/html, image/png, etc. 44
MSC	Multi-Spectral-Camera, A very-high-resolution image taking sensor on KOMPSAT-2. 2
NASA	National Aeronautics and Space Administration, established in 1958, is the agency responsible for the public space program of the United States of America. . . . 45, 75
object	In Object-Oriented Programming (OOP), an instance of a program is treated as a dynamic set of interacting objects. 68, 69
object view	In object view reality is identified by objects like houses, streets, etc.. 12
ODRL	Open Digital Rights Language 82

Notation	Description
OGC	Open Geospatial Consortium, Inc., Organization that is leading the development of standards for geospatial and location based services. iii, 8, 9, 12, 37, 38, 44, 45, 62, 63, 73, 79, 80, 82, 92, 95, 99, 100
OMGeo	Open Mobility and Geoinformation 80, 81
OpenSource	OpenSource means that the source code of the software is publicly viewable and openly modifiable.8, 45, 67, 83, 85, 87, 88, 92, 93
ORDBMS	Object-Relational Database Management System . iii, 14, 72
orthorectify	Add geospatial (geographic) information to images. 3
OS	An operating system is the set of basic programs and utilities that make your computer run. 45, 86
overview	An overview is an image at half resolution of the original image in both dimensions. This means that always four pixels get combined to one pixel in the overview, which is therefore only a quarter in size of the original image. . 34, 53
OWS	OGC Web Service . . . . . iii, 44, 45, 54, 56, 79, 94
PDF	Portable Document Format (PDF) is a file format developed by Adobe Systems for representing documents in a manner that is independent of the original application software, hardware, and operating system used to create those documents. 46, 65, 92
PDFlib	Portable C library for dynamically generating PDF files, with support for many other programming languages. . 46
pgSphere	pgSphere is an extension to PostgreSQL which provides spherical data types, functions, and operators. 19
PHP	Recursive acronym for "PHP: Hypertext Preprocessor", commonly-used OpenSource scripting language that is especially made for Web development. 8, 20, 27, 45, 46, 49, 51, 61, 63, 67–69, 92, 100
PHP/MapScript	Part of the MapServer software that allows the popular scripting language PHP to access the MapServer C API. 8, 45, 61, 63, 67–69, 100
physical schema	The physical or data schema is a fully attributed definition, detailed attributes defined for each entity, used to create a database. 13
PL/pgSQL	A loadable procedural language for the PostgreSQL database system which can be used to create functions. 20, 26, 27

Notation	Description
plugin	A plugin is a computer program that can, or must, interact with another program to provide a certain, usually very specific, function. 73, 76
PNG	Stands for Portable Network Graphics and is a raster image format with lossless compression that is popular on the WWW and elsewhere. 41
PostGIS	Extension for PostgreSQL which adds support for geographic objects. 14, 20, 45, 46, 48, 49, 53, 61, 88, 93, 99
PostgreSQL	OpenSource Database Management System. 14, 20, 45–48, 69, 92, 93, 99
primary key	One or more columns in one table, that can be used to (uniquely) identify a row in a table is called a (unique) key. One of the unique keys is the preferred way to refer to rows and is defined as the table's primary key. 21
prime meridian	Line of longitude at 0° passing through the Royal Greenwich Observatory, Greenwich, London, UK. 15
PROJ.4	Cartographic projections library used by e.g. PostGIS and MapServer for projection transformations. 16, 45
psql	A terminal-based front-end to PostgreSQL which allows to type in queries interactively. 25
raster	Raster representations underlies a grid of pixels. For every pixel the value of an attribute like elevation or land use class is saved. 12
raster-data	Raster graphics are represented by a grid of pixels each assigned individually a color. 8
raw-image	Satellite derived image with no processing, no correction applied. 2
scale	The scale of a map allows a user to measure a distance on the map and determine the distance on the ground. . . . 53
shape	Here used for geographic features, i.e. the 2 dimensional outline of satellite images or parts of them. 9
shapefile	Shapefile is a widely used file format for spatial data in geographic information systems. The format was published by ESRI in July 1998. A shapefile consists of three required files: the .shp contains geometry, the .dbf file contains attributes for each geometry and the .shx provides an index. 45

Notation	Description
SLD	Styled Layer Descriptor is an extension to allow user-defined symbolization of feature data instead of named Layers and Styles. In brief, a SLD-enabled WMS retrieves features from a WFS and applies explicit styling information provided by the user in order to render a map. . . 39, 55
SOAP	SOAP is a standard for exchanging XML-based messages over a computer network, normally using HTTP. SOAP forms the foundation layer of the web services stack, providing a basic messaging framework that more abstract layers can build on. 73, 94
spatial	Referring to some place in space, not only on Earth's surface. 1, 12
SQL	Structured Query Language, Language used to create, modify and retrieve data from relational database management systems 14, 65, 73, 74, 99, 121
SRID	Spatial Reference System Identifier, Numbering system for geographic projection systems within PostGIS. 16
SRS	The Spatial Reference System is a text parameter that names a horizontal coordinate reference system code and is used in OWSs. The name includes a namespace prefix, a colon, a numeric identifier, and potentially a comma followed by additional parameters. 44, 55, 59
SSE	Service Support Environment, Portal to Earth observation and GIS services from ESA. . . iv, 9, 10, 13, 62, 63, 71–76, 79, 80, 94, 100
TOOLBOX	The TOOLBOX is a software that has been developed for the SSE-Project. It helps the service provider to convert its service in a SOAP based service compliant with the SSE ICD 1.4 (2005). 73, 74, 76
UML	Unified Modeling Language, It is a non-proprietary, third generation modeling and specification language, used especially for object-oriented modeling. 21
UML class dia.	Graphical diagram for high-level description of logical data models. 21
UMN	University of Minnesota 45
UNIX	UNIX is a computer operating system originally developed in the 1960s and 1970s by a group of AT&T Bell Labs employees. 85, 88

Notation	Description
URL	Uniform Resource Locator, This is a standardized address name layout for resources (such as documents or images) on the Internet. 39, 40, 44, 54–56, 58–60, 63, 64, 69
UTC	Coordinated Universal Time, An atomic realization of Greenwich mean time which is the mean solar time at the Royal Greenwich Observatory in Greenwich. 21
UTM	Universal Transverse Mercator, Widely used projection system. 3, 4, 6
vector	With the vector representation objects are stored as geometrical primitives such as points, lines, polygons, and sometimes curves. 12
vector-data	Data digitally stored by identifying features as points, lines, and polygons. 8
VHR	very-high-resolution, Images with a spatial resolution less then 3 *meter* are named very-high-resolution images. . . 1, 89
watermark	Digital watermarking is a technique which allows an individual to add hidden copyright notices or other verification messages to digital audio, video, or image signals and documents. Such a hidden message is a group of bits describing information pertaining to the signal or to the author of the signal (name, place, etc.). While the addition of the hidden message to the signal does not restrict that signal's use, it provides a mechanism to track the signal to the original owner. 81, 82
WCS	WebCoverageService, OGC service to interchange geospatial data as "coverages". . . iii, 8, 32, 37, 42, 43, 45, 54, 56, 58–60, 74, 79, 99, 100, 117, 119
web browser	The program that serves as front end to the Web on the Internet. 9, 20, 27, 39, 63, 73, 76, 87
web service	According to the W3C a web service is a software system designed to support interoperable machine-to-machine interaction over a network. iii, 37, 38, 44, 45, 73, 76, 79–81, 92
WFS	WebFeatureService, This OGC service allows a client to retrieve geospatial data as features. . . iii, 8, 37, 41, 42, 45, 55, 79, 93
WGS84	World Geodetic System from 1984. . . . . 15, 53
WKB	Well-Known Binary, Form of representation of data within PostGIS. 16

Notation	Description
WKT	Well-Known Text, Form of representation of data within PostGIS. 16
WMS	WebMapService, A WMS produces maps, simple images, of georeferenced data. . iii, 8, 32, 37–45, 54–57, 59, 63–67, 69, 71–73, 76, 79, 85, 93, 99, 100, 112
WSDL	The Web Services Description Language is a XML format published for describing web services. The future version 2.0 will be a recommendation of the W3C. 73
WWW	World Wide Web, This is an information space in which the items of interest, referred to as resources, are identified by global identifiers called Uniform Resource Locators (URLs). The term is often mistakenly used as a synonym for the Internet, but the Web is actually a service that operates over the Internet. 39, 40, 73, 86, 88, 93, 96
W3C	World Wide Web Consortium, A consortium that produces the software standards ("recommendations", as the consortium calls them) for the WWW . 40, 41, 44, 70, 82, 86, 95, 96
XML	Extensible Markup Language, A W3C-recommended general-purpose markup language. A markup language combines text and extra information about the text. Its primary purpose is to facilitate the sharing of data across different systems, particularly systems connected via the Internet. . 39–42, 54, 55, 61, 70, 73, 76, 87, 88, 94, 96, 99, 100

Bibliography

Evenden G. (January 1, 2003). Cartographic Projection Procedures for the Unix Environment–A User's Manual. *U.S. Geological Survey Open-File Report 90-284*. Available at: ftp://ftp.remotesensing.org/proj/new_docs/OF90-284.pdf

ISO/TC 211 Geographic information/Geomatics (January 23, 2003). *Geographic information — Metadata (ISO/FDIS 19115:2003(E))*. Available at: http://www.ncits.org/ref-docs/FDIS_19115.pdf

Kemper A. and Eickler A. (2001). *Datenbanksysteme – Eine Einführung.* München, Wien: R. Oldenbourg Verlag, Edition: 4.

Lee E., Joo I., Kim M., and Kim M. (July 25-29, 2005). A Smart Web Platform for Telematics Services toward Ubiquitous Environments. Proceedings of the *IEEE International Geoscience And Remote Sensing Symposium, IGARSS2005* (pp. 1576-1579). Seoul, Korea.

Longley P.A., Goodchild M.F., Maguire D.J., and Rhind D.W. (2001). *Geographic Information Systems and Science.* Chichester: John Wiley & Sons, Ltd.

Open Geospatial Consortium, Inc (OGC) (n.d.). *OGC Resources - Frequently Asked Questions.* Retrieved May 17, 2005, from http://www.opengeospatial.org/resources/?page=faq

Open Geospatial Consortium, Inc (OGC) (May 5, 1999). *OpenGIS® Simple Features Specification for SQL, Revision: 1.1.* OpenGIS Project Document: 99-049. Available at: http://www.opengeospatial.org/docs/99-049.pdf

Open Geospatial Consortium, Inc (OGC) (December 12, 2000). *The OpenGIS™ Abstract Specification - Topic 11: OpenGIS™ Metadata (ISO/TC 211 DIS 19115), Version 5.* OGC™ project document: 01-111. Available at: http://www.opengeospatial.org/docs/01-111.pdf

Open Geospatial Consortium, Inc (OGC) (August 27, 2003). *Web Coverage Service (WCS), Version 1.0.0.* OpenGIS© project document: OGC 03-065r6. Available at: http://www.opengeospatial.org/docs/03-065r6.pdf

Open Geospatial Consortium, Inc (OGC) (September 19, 2002). *Web Feature Service Implementation Specification, Version: 1.0.0.* OpenGIS® project document: OGC 02-058. Available at: http://www.opengeospatial.org/docs/02-058.pdf

Open Geospatial Consortium, Inc (OGC) (January 16, 2002). *Web Map Service Implementation Specification, Version: 1.1.1.* OpenGIS® project document: OGC 01-068r3. Available at: http://www.opengeospatial.org/docs/01-068r3-.pdf

Open Geospatial Consortium, Inc (OGC) (August 2, 2004). *Web Map Service, Version: 1.3* OGC™ project document: OGC 04-024. Available at: http://portal.opengeospatial.org/files/?artifact_id=5316

Schiller C., Triebnig G., Kim Y., Ahn S., Moll B., van der Kamp A., Maass H., Schwarz J., and Kressler F. (October 27-29, 2004). KOMPSAT European Cooperation. Proceedings of the *International Symposium on Remote Sensing.* Jeju, Korea.

Schiller C., Kressler F., Triebnig G., Kim Y., Meissl S., Moll B., and Maas H. (February 22-25, 2005). European Node of the Urban Application Center for KOMPSAT-1/2 derived products. In Manfred SCHRENK (Ed.), Proceedings of the *CORP 2005 & Geomultimedia05 - 10th Int. Conf. on Information & Commun. Techn. (ICT) in Urban Planning and Spatial Development and Impacts of ICT on Physical Space* (pp. 207-212). Vienna, Austria: Competence Center of Urban and Regional Planning (CORP). Available at: http://mmp-tk1.kosnet.com/corp/archiv/papers/2005/CORP2005_SCHILLER_KRESSLER.pdf

Schiller C., Triebnig G., Kim Y., Moll B., and Maas H. (July 25-29, 2005). KOMPSAT European Cooperation. Proceedings of the *IEEE International Geoscience And Remote Sensing Symposium, IGARSS2005* (pp. 1173-1176). Seoul, Korea.

Service Support Environment (SSE), Spacebel (February 2, 2005). *MULTI APPLICATION SUPPORT SERVICE SYSTEM ENVIRONMENT — INTERFACE CONTROL DOCUMENT (ME-ICD-0001-SPB, Issue: 1, Revision: 4).* Available at: http://services.eoportal.org/massRef/documentation/icd.pdf

Teorey T.J., Yang D., and Fry J.P. (1986). A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship-Model. *ACM Computing Surveys, Vol.18, No. 2, pp.197-222.*

Wernig-Pichler J., Meissl S., Schiller C., Weinwurm G., and Triebnig G. (March 17, 2005). Use of the SSE-Toolbox and a WebMapServer to connect an EO-Catalog via the INFEO/EOLI system. *2nd SSE Workshop.* Esrin, Italy: European Space Agency. Available at: http://www.arcs.ac.at/A/publications/ESA_SSE_Workshop_March2005_Abstract.pdf

Appendix A

Source code

A.1 Introduction

This appendix provides some of the source code developed at ARC-sr. It is only an excerpt and there is a lot more required for the described system to function. The scripts, which have been added to the appendix, have been chosen to provide an overview of all tasks and for the overall functionality explained throughout the thesis. All software shown in the appendix is free software, it can be redistributed and/or modified under the terms of the GNU General Public License (GPL) as published by the Free Software Foundation. The GPL is provided in section A.12.

The following ten sections provide five scripts used for the services explained during chapters 2, 3, and 4. There is also a template configuration file (Mapfile) of the software MapServer (appendix 3.6.2), three XML documents returned by the OGC services WMS and WCS, and a sample query in SQL (appendix A.11), which can be used to develop searches in the archive provided.

The first script (appendix A.2) can be used to create the tables in a database as used for the ARCS SATdata Catalog described in section 2.4.1. If executed the script not only creates the required tables but also takes care of constraints and access privileges. The SQL data definition statements are developed for use with PostgreSQL 8.0.1, PostGIS 1.0.0, but should also work for most other versions.

The second script (appendix A.3) adds some useful functions to the database. The functions are used for administration tasks, e.g. data storage and retrieval, and are called: `insert_process()`, `insert_raw()`, `insert_geo()`, `insert_pointsfile()`, `insert_ort()`, `insert_wms()`, `select_countries()`, and `select_cities()`. The functions starting with `insert_` insert processes according to their names. The two `select_` functions are used to collect all countries or cities covered by an image into a single string.

To use these scripts a working installation of PostgreSQL and a database with PostGIS support is required. There are two ways to load the scripts into the database. The statement

```
\$ psql -f <scriptname> <database> <username>
```

can be used on the command line. The second possibility is to use the command:

```
\# \i <scriptname>
```

in the terminal-based front-end of PostgreSQL "psql".

The last script explained in chapter 2 (appendix A.4) is used to insert a "wms" process into the database. It also performs, for speed reasons, the tilling of the images and stores them in the file system. The whole process is explained in detail in section 2.4.5.

The next section 3.6.2 provides a template configuration file, called Mapfile, for the software MapServer. This Mapfile is the heart of the MapServer. It defines the relationships between objects, tells the MapServer the data location, and defines how things are to be drawn.

The following three sections A.6, A.7, and A.8 are XML documents returned by the OGC services WMS and WCS. The first one is the Capabilities XML returned from the WMS at ARC-sr after a GetCapabilities request. The second is the Capabilities XML after the same request to the WCS. The third file is a XML document describing a coverage received from a DescribeCoverage request to the WCS.

Appendix A.9 contains a PHP script called "psql2map.php". This script is used to generate a Mapfile derived from the information stored in the database called ARCS SATdata Catalog. How this script can be used is explained in section 3.8.

Another PHP script, called "makepng.php", is added in appendix A.10. This script shows how MapServer's extension PHP/MapScript can be used in the client, which is explained in chapter 4, to generate all required images, e.g. the map, the scalebar, and the reference image.

Appendix A.11 holds a template query. This query can be executed on the ARCS SATdata Catalog database. The same basic query has also been implemented in the functionality of the SSE-Portal and the eoPortal.

The last appendix (A.12) contains the GNU General Public License (GPL). All software shown in the appendix is free software, it can be redistributed and/or modified under the terms of the GPL as published by the Free Software Foundation.

A.2 ARCS_SATdata_Catalog.sql

```
-----------------------------------------------------------------
--
-- ARCS_SATdata_Catalog.sql
--
-----------------------------------------------------------------
--
-- Purpose:
--   Creates tables, constraints and privileges for the ARCS
--   SATdata Catalog. The statements are developed for use with
--   PostgreSQL 8.0.1 object-relational database management system
--   with PostGIS 1.0.0 extension for geographic objects.
--
-- Insert into database with:
--   $ psql -f ARCS_SATdata_Catalog.sql <database> <username>
--   or connect to databes with:
--     $ psql <database> [-U user]
--     and import file with:
--       # \i ARCS_SATdata_Catalog.sql
--
-----------------------------------------------------------------
--
-- Author: Stephan Meissl <smeissl@fsmat.at>
-- Copyright: Austrian Research Centers - Seibersdorf research GmbH
--
-- This is free software; you can redistribute and/or modify it
-- under the terms of hte GNU General Public Licence. See URL:
-- http://www.gnu.org/licenses/gpl.html.
--
-----------------------------------------------------------------
```

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.3 ARCS_SATdata_Catalog_Functions.sql

```
-----------------------------------------------------------------
--
-- ARCS_SATdata_Catalog_Functions.sql
--
-----------------------------------------------------------------
--
-- Purpose:
--   Creates functions for the ARCS SATdata Catalog.
--   The statements are developed for use with PostgreSQL 8.0.1
--   object-relational database management system with PostGIS
--   1.0.0 extension for geographic objects.
--
-- Insert into database with:
--   $ psql -f ARCS_SATdata_Catalog_Functions.sql <database>
--      <username>
--   or connect to databes with:
```

```
--     $ psql <database> [-U user]
--     and import file with:
--       # \i ARCS_SATdata_Catalog_Functions.sql
--
-- The functions are:
--   insert_process(VARCHAR,INTEGER,VARCHAR,VARCHAR)
--   insert_raw(VARCHAR,INTEGER,VARCHAR,VARCHAR,VARCHAR,VARCHAR)
--   insert_geo(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER)
--   insert_pointsfile(VARCHAR,INTEGER,VARCHAR,VARCHAR,VARCHAR,
     VARCHAR)
--   insert_ort(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER,VARCHAR,
--              VARCHAR,INTEGER)
--   insert_wms(VARCHAR,INTEGER,VARCHAR,VARCHAR,INTEGER,VARCHAR,
--              VARCHAR,VARCHAR,INTEGER,INTEGER,BOOLEAN)
--   select_countries(varchar)
--   select_cities(varchar)
--
-----------------------------------------------------------------
--
-- Author: Stephan Meissl <smeissl@fsmat.at>
-- Copyright: Austrian Research Centers - Seibersdorf research GmbH
--
-- This is free software; you can redistribute and/or modify it
-- under the terms of hte GNU General Public Licence. See URL:
-- http://www.gnu.org/licenses/gpl.html.
--
-----------------------------------------------------------------
```

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.4 insert_WMS_process.php

```
<?
//
// insert_WMS_process.php
//
// ------------------------------------------------------------
//
// Purpose: Process data from HTTP '$_POST' variable. Produce tiles
//          in specified location and store data about them in
//          PostgreSQL/PostGIS database (ARCS SATdata Catalog).
//          There are also recursivly tiles of half scale
//          (tileinputsize is doubled) saved  until the inputfile-
//          width or -height (the smaller one) is smaller than the
//          tileinputsize.
//
// Attention: Input needs to be and output is always GeoTiff.
//
// ------------------------------------------------------------
//
// Author: Stephan Meissl <smeissl@fsmat.at>
// Copyright: Austrian Research Centers - Seibersdorf research GmbH
//
```

```
// This is free software; you can redistribute and/or modify it
// under the terms of hte GNU General Public Licence. See URL:
// http://www.gnu.org/licenses/gpl.html.
//
// ----------------------------------------------------------------
//
```

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.5 template.map

```
MAP
  EXTENT -11 34.5 33 69.7
  IMAGECOLOR 255 255 255
  IMAGETYPE "png"
  SHAPEPATH "/home/wms_data/gis"
  SIZE 500 400
  MAXSIZE 2000
  STATUS ON
  UNITS DD
  NAME "KERA"
  FONTSET "fonts"
  DEBUG OFF

  OUTPUTFORMAT
    NAME "png"
    MIMETYPE "image/png"
    DRIVER "GD/PNG"
    EXTENSION "png"
    IMAGEMODE "RGB"
    TRANSPARENT FALSE
    FORMATOPTION "INTERLACE=ON"
  END

  OUTPUTFORMAT
    NAME "GEOTIFF"
    MIMETYPE "image/tiff"
    DRIVER "GDAL/GTiff"
    IMAGEMODE "BYTE"
    EXTENSION "tif"
  END

  SYMBOL
    NAME "cross"
    TYPE VECTOR
    FILLED TRUE
    POINTS
      5 0
      5 10
      5 5
      0 5
      10 5
      5 5
```

```
  END
END

SYMBOL
  NAME "flag"
  TYPE PIXMAP
  IMAGE "/home/wms_demo/images/flags/flag.png"
  TRANSPARENT 0
END

PROJECTION
  "init=epsg:4326"
END

LEGEND
  IMAGECOLOR 255 255 255
  KEYSIZE 20 15
  KEYSPACING 3 3
  LABEL
    ANGLE 0.000000
    ANTIALIAS TRUE
    FONT arial
    MAXSIZE 256
    MINSIZE 4
    SIZE 10
    TYPE TRUETYPE
    BUFFER 3
    COLOR 0 0 89
    FORCE FALSE
    MINDISTANCE -1
    MINFEATURESIZE -1
    OFFSET 0 0
    PARTIALS TRUE
    SHADOWSIZE 2 2
  END
  OUTLINECOLOR 255 255 255
  POSITION LC
  STATUS OFF
END

REFERENCE
  COLOR -1 -1 -1
  EXTENT -11 34.3 33.2 69.7
  IMAGE "../images/reference/europe.png"
  OUTLINECOLOR 255 0 0
  SIZE 150 120
  STATUS OFF
  MARKER "cross"
  MARKERSIZE 150
  MINBOXSIZE 20
  MAXBOXSIZE 0
END

SCALEBAR
  COLOR 255 0 0
  IMAGECOLOR 255 255 255
  INTERLACE TRUE
```

```
    INTERVALS 2
    LABEL
      SIZE TINY
      TYPE BITMAP
      BUFFER 0
      COLOR 255 0 0
      FORCE FALSE
      MINDISTANCE -1
      MINFEATURESIZE -1
      OFFSET 0 0
      PARTIALS TRUE
      SHADOWSIZE 2 2
    END
    POSITION LR
    SIZE 150 5
    STATUS OFF
    STYLE 1
    TRANSPARENT TRUE
    UNITS METERS
  END

  WEB
    IMAGEPATH "/home/wms_demo/tmp/"
    IMAGEURL ""
    LOG "/home/wms_demo/logfile"
    MAXSCALE 50000000
    METADATA
      "ows_country"             "Austria"
      "ows_keywordlist"         "KOMPSAT,EOC,EuropeanRegional,Archi
ve,KERA,Korea,KARI,ARCS,Austria"
      "ows_title""KOMPSAT European Regional Archive (KERA) at ARCS"
      "ows_city"                "Seibersdorf"
      "ows_contactperson"       "Stephan Meissl"
      "ows_addresstype"         "Office address"
      "ows_abstract"            "WMS at ARC seibersdorf research ht
tp://www.space-applications.at"
      "ows_contactposition"     "Graduant"
      "ows_onlineresource"  "http://spacey.arcs.ac.at/cgi-bin/wms?"
      "ows_stateorprovince"     "Loweraustria"
      "ows_contactorganization" "ARC seibersdorf research"
      "ows_contactelectronicmailaddress""stephan.meissl@arcs.ac.at"
      "ows_address"             "ARC seibersdorf research"
      "ows_postcode"            "2444"
      "ows_srs" "EPSG:4326 EPSG:32633 EPSG:3045 EPSG:31296"
      "ows_accessconstraints"   "none"
      "ows_fees"                "none"
      "wcs_name"                "KERA-WCS"
      "wcs_label" "WCS for the KOMPSAT European Regional Archive"
      "wcs_formats"             "GEOTIFF"
    END
    QUERYFORMAT text/html
  END

  LAYER
    METADATA
    END
    NAME "Credits"
```

```
    SIZEUNITS PIXELS
    STATUS DEFAULT
    TOLERANCEUNITS PIXELS
    TRANSFORM FALSE
    TYPE ANNOTATION
    UNITS METERS
    CLASS
      LABEL
        ANGLE 0.000000
        ANTIALIAS TRUE
        FONT arial
        MAXSIZE 256
        MINSIZE 4
        SIZE 10
        TYPE TRUETYPE
        BUFFER 2
        COLOR 0 0 0
        FORCE TRUE
        MINDISTANCE -1
        MINFEATURESIZE -1
        OFFSET 0 0
        OUTLINECOLOR 255 255 255
        PARTIALS TRUE
        POSITION LR
        SHADOWCOLOR 218 218 218
        SHADOWSIZE 2 2
      END
      METADATA
      END
    END
    FEATURE
      POINTS
        5 5
      END
      TEXT "© www.space-applications.at"
    END
  END

  LAYER
    CLASSITEM "LAYER_NAME"
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA 'geometry from (SELECT centroid(b.geometry) AS geometry, a
.layer_name FROM wms a, "GC" b WHERE b."ImageName"=a.imagename) AS
i USING UNIQUE layer_name USING SRID=4326'
    GROUP "KOMPSAT-1"
    LABELITEM "LAYER_NAME"
    METADATA
      "wms_group_title" "KOMPSAT-1"
      "wms_title"       "Marker for KOMPSAT-1 images"
    END
    NAME "kompsatmarker"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS OFF
```

```
    TOLERANCE 0
    TOLERANCEUNITS PIXELS
    TRANSPARENCY 100
    TYPE POINT
    UNITS METERS
    CLASS
      NAME "Flags for available KOMPSAT-1 data"
      LABEL
        ANGLE 0.000000
        ANTIALIAS TRUE
        FONT arial
        MAXSIZE 256
        MINSIZE 4
        SIZE 10
        TYPE TRUETYPE
        BUFFER 2
        COLOR 255 80 0
        FORCE FALSE
        MINDISTANCE -1
        MINFEATURESIZE -1
        OFFSET 0 10
        PARTIALS FALSE
        POSITION CC
        SHADOWCOLOR 218 218 218
        SHADOWSIZE 2 2
      END
      METADATA
      END
      STYLE
        ANGLE 360
        SYMBOL "flag"
      END
    END
  END

  LAYER
    CLASSITEM "NAME"
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA 'geometry from (SELECT a.wms_process_id AS oid, b.geometry
, a.layer_name AS name, c."AcqDate" AS acqdate, a.imagename AS imag
ename FROM wms a, "GC" b, image_metadata c WHERE b."ImageName"=a.im
agename AND a.imagename=c."ImageName") AS i USING UNIQUE oid USING
SRID=4326'
    GROUP "KOMPSAT-1"
    LABELITEM "NAME"
    METADATA
      "wms_group_title" "KOMPSAT-1"
      "wms_title"       "KOMPSAT-1 bounds"
    END
    NAME "kompsatoutline"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS OFF
    TOLERANCE 3
```

```
    TOLERANCEUNITS PIXELS
    TRANSPARENCY 70
    TYPE POLYGON
    UNITS METERS
    FOOTER "/home/wms_demo/html/templates/footer.html"
    HEADER "/home/wms_demo/html/templates/img_mdat_header.html"
    TEMPLATE "/home/wms_demo/html/templates/img_mdat.html"
    CLASS
      NAME "KOMPSAT-1 Bounds"
      METADATA
      END
      STYLE
        ANGLE 360
        OUTLINECOLOR 255 80 0
        SYMBOL 0
      END
    END
  END

  LAYER
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA "the_geom from (SELECT a.oid as oid, a.the_geom as the_geo
m, ('/home/ksat/wms_data/tiles/K1/eoc06182_20010216t091614/' || a.l
ocation) AS location from eoc06182_20010216t091614 a WHERE a.locati
on~'^eoc06182_20010216t091614_scale1_') as i USING UNIQUE oid USING
 SRID=4326"
    METADATA
    END
    NAME "eoc06182_20010216t091614_tileindexlayer_scale1"
    SIZEUNITS PIXELS
    STATUS OFF
    TOLERANCEUNITS PIXELS
    TYPE TILEINDEX
    UNITS METERS
  END

  LAYER
    CLASSITEM "[pixel]"
    DUMP TRUE
    GROUP "KOMPSAT-1"
    MAXSCALE 37422
    METADATA
      "ows_extent"         "16.0251 47.2766 16.8179 48.8969"
      "ows_srs"            "EPSG:4326"
      "wms_group_title"    "KOMPSAT-1"
      "wms_title"          "eoc06182_20010216T091614 Scalenr.: 1"
      "wcs_label"          "EOC 06182 February 16, 2001"
      "wcs_size"           "13380 27240"
      "wcs_bandcount"      "1"
      "wcs_nativeformat"   "8-bit GeoTIF"
      "wcs_rangeset_name"  "Panchromatic"
      "wcs_rangeset_label" "Panchromatic image from KOMPSAT-1 EOC"
    END
    NAME "eoc06182_20010216t091614_scale1"
    PROJECTION
      "init=epsg:4326"
```

```
    END
    SIZEUNITS PIXELS
    STATUS OFF
    TILEINDEX "eoc06182_20010216t091614_tileindexlayer_scale1"
    TILEITEM "location"
    TOLERANCEUNITS PIXELS
    TYPE RASTER
    UNITS METERS
    CLASS
      EXPRESSION ([pixel] > 0)
      METADATA
      END
    END
  END

  LAYER
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA "the_geom from (SELECT a.oid as oid, a.the_geom as the_geo
m, ('/home/ksat/wms_data/tiles/K1/eoc06182_20010216t091614/' || a.l
ocation) AS location from eoc06182_20010216t091614 a WHERE a.locati
on~'^eoc06182_20010216t091614_scale2_') as i USING UNIQUE oid USING
 SRID=4326"
    METADATA
    END
    NAME "eoc06182_20010216t091614_tileindexlayer_scale2"
    SIZEUNITS PIXELS
    STATUS OFF
    TOLERANCEUNITS PIXELS
    TYPE TILEINDEX
    UNITS METERS
  END

  LAYER
    CLASSITEM "[pixel]"
    GROUP "KOMPSAT-1"
    MAXSCALE 74844
    METADATA
      "ows_extent"      "16.0251 47.2766 16.8179 48.8969"
      "ows_srs"         "EPSG:4326"
      "wms_group_title" "KOMPSAT-1"
      "wms_title"       "eoc06182_20010216T091614 Scalenr.: 2"
    END
    MINSCALE 37423
    NAME "eoc06182_20010216t091614_scale2"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS OFF
    TILEINDEX "eoc06182_20010216t091614_tileindexlayer_scale2"
    TILEITEM "location"
    TOLERANCEUNITS PIXELS
    TYPE RASTER
    UNITS METERS
    CLASS
      EXPRESSION ([pixel] > 0)
```

```
      METADATA
      END
    END
  END

  LAYER
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA "the_geom from (SELECT a.oid as oid, a.the_geom as the_geo
m, ('/home/ksat/wms_data/tiles/K1/eoc06182_20010216t091614/' || a.l
ocation) AS location from eoc06182_20010216t091614 a WHERE a.locati
on~'^eoc06182_20010216t091614_scale3_') as i USING UNIQUE oid USING
 SRID=4326"
    METADATA
    END
    NAME "eoc06182_20010216t091614_tileindexlayer_scale3"
    SIZEUNITS PIXELS
    STATUS OFF
    TOLERANCEUNITS PIXELS
    TYPE TILEINDEX
    UNITS METERS
  END

  LAYER
    CLASSITEM "[pixel]"
    GROUP "KOMPSAT-1"
    MAXSCALE 149688
    METADATA
      "ows_extent"      "16.0251 47.2766 16.8179 48.8969"
      "ows_srs"         "EPSG:4326"
      "wms_group_title" "KOMPSAT-1"
      "wms_title"       "eoc06182_20010216T091614 Scalenr.: 3"
    END
    MINSCALE 74845
    NAME "eoc06182_20010216t091614_scale3"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS OFF
    TILEINDEX "eoc06182_20010216t091614_tileindexlayer_scale3"
    TILEITEM "location"
    TOLERANCEUNITS PIXELS
    TYPE RASTER
    UNITS METERS
    CLASS
      EXPRESSION ([pixel] > 0)
      METADATA
      END
    END
  END

  LAYER
    CONNECTION "user=wmsuser dbname=ksat host=/tmp"
    CONNECTIONTYPE POSTGIS
    DATA "the_geom from (SELECT a.oid as oid, a.the_geom as the_geo
m, ('/home/ksat/wms_data/tiles/K1/eoc06182_20010216t091614/' || a.l
```

```
ocation) AS location from eoc06182_20010216t091614 a WHERE a.locati
on~'^eoc06182_20010216t091614_scale4_') as i USING UNIQUE oid USING
 SRID=4326"
    METADATA
    END
    NAME "eoc06182_20010216t091614_tileindexlayer_scale4"
    SIZEUNITS PIXELS
    STATUS OFF
    TOLERANCEUNITS PIXELS
    TYPE TILEINDEX
    UNITS METERS
  END

  LAYER
    CLASSITEM "[pixel]"
    GROUP "KOMPSAT-1"
    METADATA
      "ows_extent"      "16.0251 47.2766 16.8179 48.8969"
      "ows_srs"         "EPSG:4326"
      "wms_group_title" "KOMPSAT-1"
      "wms_title"       "eoc06182_20010216T091614 Scalenr.: 4"
    END
    MINSCALE 149689
    NAME "eoc06182_20010216t091614_scale4"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS OFF
    TILEINDEX "eoc06182_20010216t091614_tileindexlayer_scale4"
    TILEITEM "location"
    TOLERANCEUNITS PIXELS
    TYPE RASTER
    UNITS METERS
    CLASS
      EXPRESSION ([pixel] > 0)
      METADATA
      END
    END
  END

  LAYER
    MAXSCALE 38000
    METADATA
    END
    NAME "zoomgrid"
    PROJECTION
      "init=epsg:4326"
    END
    SIZEUNITS PIXELS
    STATUS DEFAULT
    TOLERANCEUNITS PIXELS
    TYPE LINE
    UNITS METERS
    CLASS
      NAME "zoomgrid"
      LABEL
```

```
        ANGLE 0.000000
        ANTIALIAS TRUE
        FONT arial
        MAXSIZE 256
        MINSIZE 4
        SIZE 8
        TYPE TRUETYPE
        BUFFER 5
        COLOR 0 0 0
        FORCE FALSE
        MINDISTANCE 200
        MINFEATURESIZE -1
        OFFSET 0 0
        OUTLINECOLOR 255 255 255
        PARTIALS FALSE
        POSITION AUTO
      END
      METADATA
      END
      STYLE
        ANGLE 360
        COLOR 0 0 0
        SYMBOL 0
      END
    END
    GRID
      MINARCS 10
      MAXARCS 10
      LABELFORMAT "DDMMSS"
    END
  END

END
```

A.6 WMS GetCapabilities.xml

```
<?xml version='1.0' encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE WMT_MS_Capabilities SYSTEM "http://schemas.opengeospatial
.net/wms/1.1.1/capabilities_1_1_1.dtd"
 [
 <!ELEMENT VendorSpecificCapabilities EMPTY>
 ]>  <!-- end of DOCTYPE declaration -->

<WMT_MS_Capabilities version="1.1.1">

<!-- MapServer version 4.6.0 OUTPUT=GIF OUTPUT=PNG OUTPUT=JPEG OUTP
UT=WBMP OUTPUT=PDF OUTPUT=SVG SUPPORTS=PROJ SUPPORTS=FREETYPE SUPPO
RTS=WMS_SERVER SUPPORTS=WMS_CLIENT SUPPORTS=WFS_SERVER SUPPORTS=WFS
_CLIENT SUPPORTS=WCS_SERVER SUPPORTS=GEOS INPUT=EPPL7 INPUT=POSTGIS
INPUT=OGR INPUT=GDAL INPUT=SHAPEFILE DEBUG=MSDEBUG -->

<Service>
  <Name>OGC:WMS</Name>
  <Title>KOMPSAT European Regional Archive (KERA) at ARCS</Title>
  <Abstract>WMS at ARC seibersdorf research http://www.space-applic
```

```
  ations.at</Abstract>
        <KeywordList>
          <Keyword>KOMPSAT</Keyword>
          <Keyword>EOC</Keyword>
          <Keyword>European</Keyword>
          <Keyword>Regional</Keyword>
          <Keyword>Archive</Keyword>
          <Keyword>KERA</Keyword>
          <Keyword>Korea</Keyword>
          <Keyword>KARI</Keyword>
          <Keyword>ARCS</Keyword>
          <Keyword>Austria</Keyword>
        </KeywordList>
  <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:
href="http://spacey.arcs.ac.at/cgi-bin/wms?"/>
  <ContactInformation>
    <ContactPersonPrimary>
      <ContactPerson>Stephan Meissl</ContactPerson>
      <ContactOrganization>ARC seibersdorf research</ContactOrganiz
ation>
    </ContactPersonPrimary>
      <ContactPosition>Graduant</ContactPosition>
    <ContactAddress>
        <AddressType>Office address</AddressType>
        <Address>ARC seibersdorf research</Address>
        <City>Seibersdorf</City>
        <StateOrProvince>Loweraustria</StateOrProvince>
        <PostCode>2444</PostCode>
        <Country>Austria</Country>
    </ContactAddress>
  <ContactElectronicMailAddress>stephan.meissl@arcs.ac.at</ContactE
lectronicMailAddress>
  </ContactInformation>
  <Fees>none</Fees>
  <AccessConstraints>none</AccessConstraints>
</Service>

<Capability>
  <Request>
    <GetCapabilities>
      <Format>application/vnd.ogc.wms_xml</Format>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/
xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Get>
          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999
/xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Post>
        </HTTP>
      </DCPType>
    </GetCapabilities>
    <GetMap>
      <Format>image/png</Format>
      <Format>image/tiff</Format>
      <Format>image/gif</Format>
      <Format>image/png; mode=24bit</Format>
      <Format>image/jpeg</Format>
      <Format>image/wbmp</Format>
```

```
      <DCPType>
        <HTTP>
          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/
xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Get>
          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999
/xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Post>
        </HTTP>
      </DCPType>
    </GetMap>
    <GetFeatureInfo>
      <Format>text/plain</Format>
      <Format>application/vnd.ogc.gml</Format>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/
xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Get>
          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999
/xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Post>
        </HTTP>
      </DCPType>
    </GetFeatureInfo>
    <DescribeLayer>
      <Format>text/xml</Format>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/
xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Get>
          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999
/xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Post>
        </HTTP>
      </DCPType>
    </DescribeLayer>
    <GetLegendGraphic>
      <Format>image/png</Format>
      <Format>image/gif</Format>
      <Format>image/png; mode=24bit</Format>
      <Format>image/jpeg</Format>
      <Format>image/wbmp</Format>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/
xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Get>
          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999
/xlink" xlink:href="http://spacey.arcs.ac.at/cgi-bin/wms?"/></Post>
        </HTTP>
      </DCPType>
    </GetLegendGraphic>
  </Request>
  <Exception>
    <Format>application/vnd.ogc.se_xml</Format>
    <Format>application/vnd.ogc.se_inimage</Format>
    <Format>application/vnd.ogc.se_blank</Format>
  </Exception>
  <VendorSpecificCapabilities />
  <UserDefinedSymbolization SupportSLD="1" UserLayer="0" UserStyle=
"1" RemoteWFS="0"/>
  <Layer>
```

```
    <Name>KERA</Name>
    <Title>KOMPSAT European Regional Archive (KERA) at ARCS</Title>
    <SRS>EPSG:4326</SRS>
    <SRS>EPSG:32633</SRS>
    <SRS>EPSG:3045</SRS>
    <SRS>EPSG:31296</SRS>
    <LatLonBoundingBox minx="-11" miny="34.5" maxx="33" maxy="69.7"
 />
    <BoundingBox SRS="EPSG:4326"
                minx="-11" miny="34.5" maxx="33" maxy="69.7" />
    <ScaleHint min="0" max="24945.1" />
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>Credits</Name>
        <Title>Credits</Title>
    </Layer>
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>zoomgrid</Name>
        <Title>zoomgrid</Title>
        <SRS>EPSG:4326</SRS>
        <LatLonBoundingBox minx="0" miny="0" maxx="0" maxy="0" />
        <BoundingBox SRS="EPSG:4326"
                    minx="0" miny="0" maxx="0" maxy="0" />
        <Style>
          <Name>default</Name>
          <Title>default</Title>
          <LegendURL width="20" height="15">
             <Format>image/png</Format>
             <OnlineResource xmlns:xlink="http://www.w3.org/1999/xl
ink" xlink:type="simple" xlink:href="http://spacey.arcs.ac.at/cgi-b
in/wms?version=1.1.1&amp;service=WMS&amp;request=GetLegendGraphic&a
mp;layer=zoomgrid&amp;format=image/png"/>
          </LegendURL>
        </Style>
        <ScaleHint min="0" max="18.9583" />
    </Layer>
    <Layer>
      <Name>KOMPSAT-1</Name>
      <Title>KOMPSAT-1</Title>
      <Abstract>KOMPSAT-1</Abstract>
      <Layer queryable="0" opaque="0" cascaded="0">
        <Name>kompsatmarker</Name>
        <Title>Marker for KOMPSAT-1 images</Title>
        <SRS>EPSG:4326</SRS>
        <Style>
          <Name>default</Name>
          <Title>default</Title>
          <LegendURL width="20" height="15">
             <Format>image/png</Format>
             <OnlineResource xmlns:xlink="http://www.w3.org/1999/xl
ink" xlink:type="simple" xlink:href="http://spacey.arcs.ac.at/cgi-b
in/wms?version=1.1.1&amp;service=WMS&amp;request=GetLegendGraphic&a
mp;layer=kompsatmarker&amp;format=image/png"/>
          </LegendURL>
        </Style>
      </Layer>
      <Layer queryable="1" opaque="0" cascaded="0">
        <Name>kompsatoutline</Name>
```

```
        <Title>KOMPSAT-1 bounds</Title>
        <SRS>EPSG:4326</SRS>
        <Style>
          <Name>default</Name>
          <Title>default</Title>
          <LegendURL width="20" height="15">
             <Format>image/png</Format>
             <OnlineResource xmlns:xlink="http://www.w3.org/1999/xl
ink" xlink:type="simple" xlink:href="http://spacey.arcs.ac.at/cgi-b
in/wms?version=1.1.1&amp;service=WMS&amp;request=GetLegendGraphic&a
mp;layer=kompsatoutline&amp;format=image/png"/>
          </LegendURL>
        </Style>
      </Layer>
      <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_scale1</Name>
        <Title>eoc06182_20010216T091614 Scalenr.: 1</Title>
        <SRS>EPSG:4326</SRS>
        <LatLonBoundingBox minx="16.0251" miny="47.2766" maxx="16.8
179" maxy="48.8969" />
        <BoundingBox SRS="EPSG:4326"
                     minx="16.0251" miny="47.2766" maxx="16.8179" ma
xy="48.8969" />
        <ScaleHint min="0" max="18.6699" />
      </Layer>
      <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_scale2</Name>
        <Title>eoc06182_20010216T091614 Scalenr.: 2</Title>
        <SRS>EPSG:4326</SRS>
        <LatLonBoundingBox minx="16.0251" miny="47.2766" maxx="16.8
179" maxy="48.8969" />
        <BoundingBox SRS="EPSG:4326"
                     minx="16.0251" miny="47.2766" maxx="16.8179" ma
xy="48.8969" />
        <ScaleHint min="18.6704" max="37.3399" />
      </Layer>
      <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_scale3</Name>
        <Title>eoc06182_20010216T091614 Scalenr.: 3</Title>
        <SRS>EPSG:4326</SRS>
        <LatLonBoundingBox minx="16.0251" miny="47.2766" maxx="16.8
179" maxy="48.8969" />
        <BoundingBox SRS="EPSG:4326"
                     minx="16.0251" miny="47.2766" maxx="16.8179" ma
xy="48.8969" />
        <ScaleHint min="37.3404" max="74.6798" />
      </Layer>
      <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_scale4</Name>
        <Title>eoc06182_20010216T091614 Scalenr.: 4</Title>
        <SRS>EPSG:4326</SRS>
        <LatLonBoundingBox minx="16.0251" miny="47.2766" maxx="16.8
179" maxy="48.8969" />
        <BoundingBox SRS="EPSG:4326"
                     minx="16.0251" miny="47.2766" maxx="16.8179" ma
xy="48.8969" />
        <ScaleHint min="74.6803" max="0" />
```

```
        <!-- WARNING: Only MINSCALE and no MAXSCALE specified in th
e mapfile. A default value of 0 has been returned for the Max Scale
Hint but this is probably not what you want. -->
      </Layer>
    </Layer>
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_tileindexlayer_scale1</Name>
        <Title>eoc06182_20010216t091614_tileindexlayer_scale1</Titl
e>
    </Layer>
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_tileindexlayer_scale2</Name>
        <Title>eoc06182_20010216t091614_tileindexlayer_scale2</Titl
e>
    </Layer>
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_tileindexlayer_scale3</Name>
        <Title>eoc06182_20010216t091614_tileindexlayer_scale3</Titl
e>
    </Layer>
    <Layer queryable="0" opaque="0" cascaded="0">
        <Name>eoc06182_20010216t091614_tileindexlayer_scale4</Name>
        <Title>eoc06182_20010216t091614_tileindexlayer_scale4</Titl
e>
    </Layer>
  </Layer>
</Capability>
</WMT_MS_Capabilities>
```

A.7 WCS GetCapabilities.xml

```
<?xml version='1.0' encoding="ISO-8859-1" standalone="no" ?>
<WCS_Capabilities
   version="1.0.0"
   updateSequence="0"
   xmlns="http://www.opengis.net/wcs"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wcs http://schemas.op
engeospatial.net/wcs/1.0.0/wcsCapabilities.xsd">
<Service>
  <name>KERA-WCS</name>
  <label>WCS for the KOMPSAT European Regional Archive</label>
  <keywords>
    <keyword>KOMPSAT</keyword>
    <keyword>EOC</keyword>
    <keyword>European</keyword>
    <keyword>Regional</keyword>
    <keyword>Archive</keyword>
    <keyword>KERA</keyword>
    <keyword>Korea</keyword>
    <keyword>KARI</keyword>
    <keyword>ARCS</keyword>
    <keyword>Austria</keyword>
```

```
  </keywords>
<responsibleParty>
    <individualName>Stephan Meissl</individualName>
    <organisationName>ARC seibersdorf research</organisationName>
    <positionName>Graduant</positionName>
  <contactInfo>
    <address>
    <deliveryPoint>ARC seibersdorf research</deliveryPoint>
    <city>Seibersdorf</city>
    <administrativeArea>Loweraustria</administrativeArea>
    <postalCode>2444</postalCode>
    <country>Austria</country>
    <electronicMailAddress>stephan.meissl@arcs.ac.at</electronicMai
lAddress>
    </address>
  </contactInfo>
</responsibleParty>
  <fees>none</fees>
  <accessConstraints>
    none
  </accessConstraints>
</Service>
<Capability>
  <Request>
    <GetCapabilities>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xlink:type="simple" xlink:href="http
://spacey.arcs.ac.at/cgi-bin/wms?" /></Get>
        </HTTP>
      </DCPType>
      <DCPType>
        <HTTP>
          <Post><OnlineResource xlink:type="simple" xlink:href="htt
p://spacey.arcs.ac.at/cgi-bin/wms?" /></Post>
        </HTTP>
      </DCPType>
    </GetCapabilities>
    <DescribeCoverage>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xlink:type="simple" xlink:href="http
://spacey.arcs.ac.at/cgi-bin/wms?" /></Get>
        </HTTP>
      </DCPType>
      <DCPType>
        <HTTP>
          <Post><OnlineResource xlink:type="simple" xlink:href="htt
p://spacey.arcs.ac.at/cgi-bin/wms?" /></Post>
        </HTTP>
      </DCPType>
    </DescribeCoverage>
    <GetCoverage>
      <DCPType>
        <HTTP>
          <Get><OnlineResource xlink:type="simple" xlink:href="http
://spacey.arcs.ac.at/cgi-bin/wms?" /></Get>
```

```
        </HTTP>
      </DCPType>
      <DCPType>
        <HTTP>
          <Post><OnlineResource xlink:type="simple" xlink:href="htt
p://spacey.arcs.ac.at/cgi-bin/wms?" /></Post>
        </HTTP>
      </DCPType>
    </GetCoverage>
  </Request>
  <Exception>
    <Format>application/vnd.ogc.se_xml</Format>
  </Exception>
  <VendorSpecificCapabilities />
</Capability>
<ContentMetadata>
  <CoverageOfferingBrief>
  <name>eoc06182_20010216t091614_scale1</name>
  <label>EOC 06182 February 16, 2001</label>
    <lonLatEnvelope srsName="WGS84(DD)">
      <gml:pos>16.0251 47.2766</gml:pos>
      <gml:pos>16.8179 48.8969</gml:pos>
    </lonLatEnvelope>
  </CoverageOfferingBrief>
</ContentMetadata>
</WCS_Capabilities>
```

A.8 WCS DescribeCoverage.xml

```
<?xml version='1.0' encoding="ISO-8859-1" ?>
<CoverageDescription
   version="1.0.0"
   updateSequence="0"
   xmlns="http://www.opengis.net/wcs"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wcs http://schemas.op
engeospatial.net/wcs/1.0.0/describeCoverage.xsd">
  <CoverageOffering>
  <name>eoc06182_20010216t091614_scale1</name>
  <label>EOC 06182 February 16, 2001</label>
    <lonLatEnvelope srsName="WGS84(DD)">
      <gml:pos>16.0251 47.2766</gml:pos>
      <gml:pos>16.8179 48.8969</gml:pos>
    </lonLatEnvelope>
    <domainSet>
      <spatialDomain>
        <gml:Envelope srsName="WGS84(DD)">
          <gml:pos>16.0251 47.2766</gml:pos>
          <gml:pos>16.8179 48.8969</gml:pos>
        </gml:Envelope>
        <gml:Envelope srsName="EPSG:4326">
          <gml:pos>16.0251 47.2766</gml:pos>
          <gml:pos>16.8179 48.8969</gml:pos>
```

```
        </gml:Envelope>
        <gml:RectifiedGrid dimension="2">
          <gml:limits>
            <gml:GridEnvelope>
              <gml:low>0 0</gml:low>
              <gml:high>13379 27239</gml:high>
            </gml:GridEnvelope>
          </gml:limits>
          <gml:axisName>x</gml:axisName>
          <gml:axisName>y</gml:axisName>
          <gml:origin>
            <gml:pos>16.0251 48.8969</gml:pos>
          </gml:origin>
          <gml:offsetVector>5.92526158445443e-05 0</gml:offsetVecto
r>
          <gml:offsetVector>0 -5.94823788546256e-05</gml:offsetVect
or>
        </gml:RectifiedGrid>
      </spatialDomain>
    </domainSet>
    <rangeSet>
      <RangeSet>
        <name>Panchromatic</name>
        <label>Panchromatic image from KOMPSAT-1 EOC</label>
      </RangeSet>
    </rangeSet>
    <supportedCRSs>
      <requestResponseCRSs>EPSG:4326</requestResponseCRSs>
      <nativeCRSs>EPSG:4326</nativeCRSs>
    </supportedCRSs>
    <supportedFormats nativeFormat="8-bit GeoTIFF">
      <formats>GeoTIFF</formats>
    </supportedFormats>
  </CoverageOffering>
</CoverageDescription>
```

A.9 psql2map.php

```
#!/usr/local/bin/php -C
<?
//
// psql2map.php, Version 3.0, 20050726
//
// ------------------------------------------------------------------
//
// Purpose: Look for 'wms' processes in the database (ARCS SATdata
//          Catalog) '$database' and store found information in
//          mapfile '$mapfile_output' using '$mapfile_templ' as a
//          template.
//          Specified user needs read privileges to the database!
//
// ------------------------------------------------------------------
//
// Author: Stephan Meissl <smeissl@fsmat.at>
// Copyright: Austrian Research Centers - Seibersdorf research GmbH
```

```
//
// This is free software; you can redistribute and/or modify it
// under the terms of hte GNU General Public Licence. See URL:
// http://www.gnu.org/licenses/gpl.html.
//
// ----------------------------------------------------------------
//
// Usage: psql2map.php database user mapfile_templ mapfile_output
//
// ----------------------------------------------------------------
//
```

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.10 bin/makepng.php

```
<?
//
// bin/makepng.php
//
// ----------------------------------------------------------------
//
// Purpose: Creates new images based on information from HTTP
//          form '$_POST' variables and stores the URLs in the
//          session variable.
//
// ----------------------------------------------------------------
//
// Author: Stephan Meissl <stephan.meissl@arcs.ac.at>
// Copyright: Austrian Research Centers - Seibersdorf research GmbH
//
// This is free software; you can redistribute and/or modify it
// under the terms of hte GNU General Public Licence. See URL:
// http://www.gnu.org/licenses/gpl.html.
//
// ----------------------------------------------------------------
//
```

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.11 SQL query

The full source code is available at ARC Seibersdorf research GmbH. For more information please contact Stephan Meißl at smeissl@fsmat.at.

A.12 The GNU General Public License (GPL)

Version 2, June 1991

51 Franklin St, Fifth Floor, Boston, MA 02110-1301, USA

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to

know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU General Public License

Terms and Conditions For Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

(a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

(b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

(c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

(a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of

the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are

imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for

permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

End of Terms and Conditions

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> <one line to give the program's name and a brief idea of what it does.>
> Copyright (C) <year> <name of author>
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) <year> <name of author>
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
> This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program
> 'Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> <signature of Ty Coon>, 1 April 1989
> Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.