UNIVERSITY OF CAPE TOWN

Web-based Management of Time-series Raster Data

A dissertation submitted to the Department of Computer Science, Faculty of Science, at the University of Cape Town, in partial fulfilment of the requirements for the degree of Master of Science (in Information Technology).

By Blessing Siwela
June 2010

Supervised by Professor Sonia Berman

Abstract

Data discovery and data handling often presents serious challenges to organizations that manage huge archives of raster datasets such as those generated by satellite remote sensing. Satellite remote sensing produces a regular stream of raster datasets used in many applications including environmental and agricultural monitoring. This thesis presents a system architecture for the management of time-series GIS raster datasets. The architecture is then applied in a prototype implementation for a department that uses remote sensing data for agricultural monitoring. The architecture centres on three key components.

The first is a metadatabase to hold metadata for the raster datasets, and an interface to manage the metadatabase and facilitate the search and discovery of raster metadata. The design of the metadatabase involved the examination of existing standards for geographic raster metadata and the determination of the metadata elements required for time-series raster data. The second component is an interactive tool for viewing the time-series raster data discovered via the metadatabase. The third component provides basic image analysis functionality typically required by users of time-series raster datasets.

A prototype was implemented using open source software and following the Open Geospatial Consortium specifications for web map services (WMS) version 1.3.0. After implementation, an evaluation of the prototype was carried out by the target users from the RRSU (Regional Remote Sensing Unit) to assess the usability, the added value of the prototype and its impact on the work of the users.

The evaluation showed that the prototype system was generally well received, since it allowed both the data managers and users of time-series datasets to save significant amounts of time in their work routines and it also offered some raster data analyses that are useful to a wider community of time-series raster data managers.

Acknowledgements

A sincere word of gratitude goes to Prof. Sonia Berman for supervising this research. The author is deeply indebted to her for her time, useful insights and advice throughout the execution of this research.

Many thanks also go to my colleagues from the SADC Regional Remote Sensing Unit (RRSU) for their active participation in the development of the prototype system and their time in evaluating the prototype. They provided useful insights on what they expected from the research and also offered invaluable moral support throughout my research. I would like to thank Mr Khrishan Bheenick for providing useful comments on the interface of the prototype system.

The quality of this write-up also benefited from comments supplied by friends and workmates who took their time to review the write-up.

I would also like to thank fellow UCT colleagues for their morale support. I would especially like to convey a word of thanks to Mr. Denis Cheruiyot for his support and continuing to check on my progress while I conducted this research.

Last, but certainly not least, I would like to convey a special word of thanks to my family for their moral support, encouragement and prayers.

Table of Contents

List of Tables

List of Figures

Chapter 1 - Introduction

This chapter outlines the background and motivation for this research and presents the research questions and objectives. It also presents the outline of the research.

1.1 Background and motivation

Satellite remote sensing is a cost-effective and powerful data acquisition method which provides raster data for a wide range of applications. Many organizations collect and use these datasets in areas including environmental monitoring, natural resources management, agriculture and weather monitoring. Large amounts of raster data are being gathered continuously and data archives continue to grow.

As the data archives grow, users face the challenge of discovering desired datasets and retrieving information from them. This mainly results from difficulties in implementing raster data archives in ways that promote efficient discovery. There is generally inadequate database support for the storage of raster data and the result is that most raster data archives are implemented as file-based storage systems [6, 31]. These file-based implementations suffer from lack of support for querying multiple raster datasets, particularly in a multi-user environment. Queries on the raster data are usually performed on a file by file basis, using specialised raster data analysis software. To accomplish a query on one raster data file, users usually perform the following subtasks — discovering the appropriate raster data file, opening the file using the appropriate software, performing the query, and discovering the results of the query. Each of these subtasks is performed with a level of difficulty determined by the user-friendliness of the interfaces used to perform it. Most of the readily available raster data handling software lack user friendly means for performing these tasks.

Raster or image data processing software generally lack data management capabilities [5]. This leaves the users with the task of managing the raster data separately and data discovery tends to be tedious and prone to error. Discovering data or files in file-based archives demands sound knowledge of the archive set-up and the file naming conventions.

Raster data from different sources are usually provided in different formats and this means that users need a variety of decoders to be able to read these image formats, and the users may not always know how to use the required decoding software. Furthermore, this means that the users need to acquire an array of the required software tools and this is not trivial task due to a variety of reasons, including software licensing. Some open source libraries are available which allow developers to read many image formats but these are not usually packaged in a user friendly manner.

Raster data analysis tools tend to require significant expertise to use and usually involve specialised software packages. User friendly data analysis tools are not readily available, particularly those that can be accessed on the Internet. Available analysis tools seldom provide functionality to handle time-series raster data as is required by users in the fields of environmental and agriculture monitoring. These users tend to require functionality that allows them to perform some algebra or statistics on the raster data. For example, a common practice in agricultural season monitoring involves the comparison of a given period of the current year with the same period of the past year or with mean (or average) conditions for the period, which allows a reasonable estimation of the quality of the current season [24]. In preparation for this, these users also often need to create averages of time-series raster data.

Most of the readily available analysis tools also provide limited opportunities for raster data users to automate routine tasks which could save them time in the preparation of products which depend on the results of the raster data analysis. Users in the environment and agriculture field often produce state of the environment or crop status reports that rely heavily on the analysis of time-series raster data and a lot of time is spent on analysing the raster data.

Many users work in distributed environments nowadays and this often creates the need to access data resources held on other computers. This is more so for users of raster data since raster data archives are usually kept on powerful file servers because of the large file sizes involved. This distributed access to raster data resources is generally hampered by the lack of web-based systems which offer the kind of analysis functionality required.

1.2 Objectives

This research aims to determine the metadata and functionality required for discovery and routine analysis of time-series raster data, and to propose a system architecture for such a system. The first objective is thus to propose a schema for time-series environmental raster data metadata which will form the basis for the implementation of a metadatabase for the raster data.

The second objective is to propose a system architecture and suite of functions for easy and effective use of raster data. The system will address raster data discovery, web-based data viewing and some basic time-series image analysis functionality.

In order to assess the viability and success of the proposed approach, the third objective is to build a prototype implementation for the Regional Remote Sensing Unit (RRSU) using free and open source technologies. The RRSU is a department that uses environmental raster data obtained from satellite sensors to monitor the status of the agriculture in southern Africa. The department is based at the Southern Africa Development Community (SADC) headquarters in Gaborone, operating within the Food Agriculture and Natural Resources (FANR) directorate. Agriculture remains the most vital sector in the economies of most countries in southern Africa. One of the main activities of the RRSU is the monitoring of rainfall, vegetation and crop performance and preparation of regular reports on the resultant implications on agriculture and food security. The regular reports or bulletins produced usually include maps and statistics (presented in tables or graphs) derived from satellite image datasets (examples in figure 1.1).

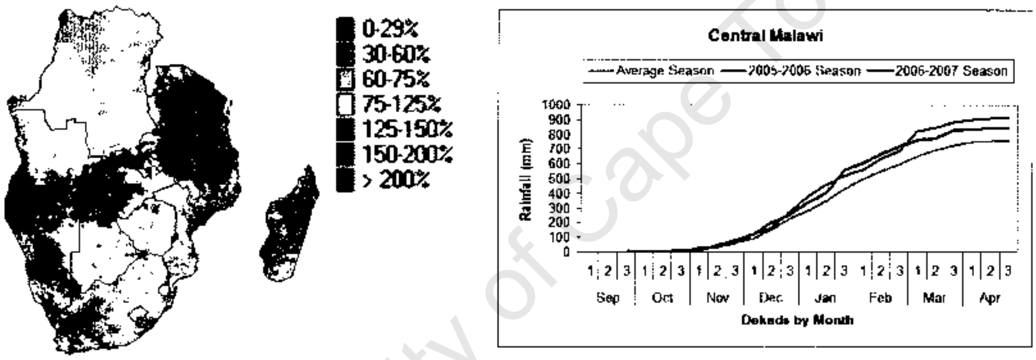


Figure 1.1: Example of products derived from satellite raster data (imagery).

Map in figure 1.1 shows rainfall estimates compared with average while the graph shows rainfall estimates extracted from a time-series of satellite images for a selected zone. The regular monitoring allows the RRSU to issue early warnings of potential food security problems arising from difficulties faced in agriculture.

The success of the RRSU in discharging their monitoring duties largely depends on the ability to properly manage their raster data archives. Currently, users at RRSU face serious challenges with data discovery and raster data analysis.

The final objective of this research is to evaluate, with the help of the users, the prototype system to understand its usability, strengths and weaknesses.

1.3 Dissertation Outline

Chapter 1 presented the motivation and objectives of this research.

Chapter 2 provides a literature review of aspects relevant to the research, covering metadata management for geographic raster datasets, web-based viewing of raster data and image analysis.

Chapter 3 presents design considerations for a time-series raster data management system and then presents a proposed system architecture, along with the decisions that were taken in the development of this proposal.

Chapter 4 presents the design of the metadatabase, including the proposed metadata schema and the associated data dictionary.

Chapter 5 presents the prototype and its interface that allows users to interact with the metadatabase management system, the web mapping service and the image analysis functionality.

Chapter 6 presents the implementation of the prototype system including the metadatabase, web map service and image analysis functionality.

Chapter 7 presents the steps taken to evaluate the prototype system and the results of the evaluation.

Chapter 8 presents conclusions and recommendations.

Chapter 2 - Background

This chapter presents the main concepts relevant to the handling of raster datasets, including an introduction to raster data, metadata, metadata standards and metadata management approaches. The chapter also presents an overview of web-based GIS concepts and image analysis functionality.

2.1 Raster Data

In the field of Geographic Information Systems, the term raster is used to refer to a data model in which the locations of geographic objects or conditions are defined by the column and row position of the cells that they occupy in a two-dimension grid [27]. The value stored for each grid cell represents the type of object or condition that prevails at the location.
Digital images, which are pictorial representations of photographic or other data, are usually presented in this format and are often more appropriately referred to as raster images [31].

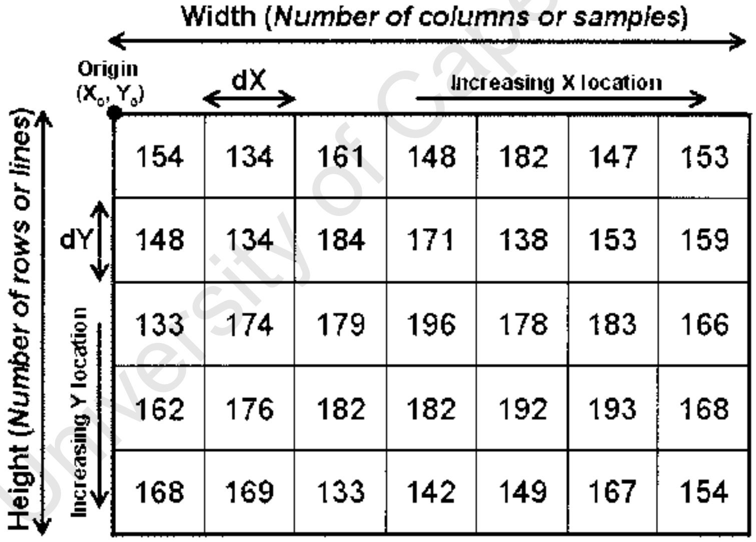


Figure 2.1 (a): Raster data model

Figure 2.1 shows the raster data model. Each of the grid cells represents an area of size dX x dY on the ground. The coordinate system of the grid usually starts at top left position (0,0) with X position increasing towards the right and Y position increasing towards the bottom of the grid [27, 31]. The correct ground position of the origin (Xo, Yo) is used to correctly position (geo-reference) the raster data, that is, to give all the grid cells the positions that they represent on the ground.

The display of raster data is usually achieved via the use of colour tables or look-up tables which tells the displaying software which colours to use for each of the values in the raster or grid [31]. Figure 2.1 (b) shows the result of applying a colour table on the raster data presented in figure 2.1(a) to get pictorial representation of the raster data. For example, all raster cell values between 0 and 40 are given the colour RGB(255, 231, 205). This representation, when used in conjunction with an appropriate legend, makes it easier for people to interpret the data.

Colour table

From	To	Red	Green	Blue	Color
0	40	255	231	205	
41	54	255	255	200	
55	62	220	230	100	
63	68	230	230	50	
69	76	200	255	150	
77	86	200	255	100	
87	100	120	255	0	
101	116	120	230	0	
117	136	120	200	0	
137	155	0	175	0	
156	186	0	140	0	
187	197	0	100	0	
198	216	0	75	0	
217	253	0	50	0	
254	255	192	192	192	

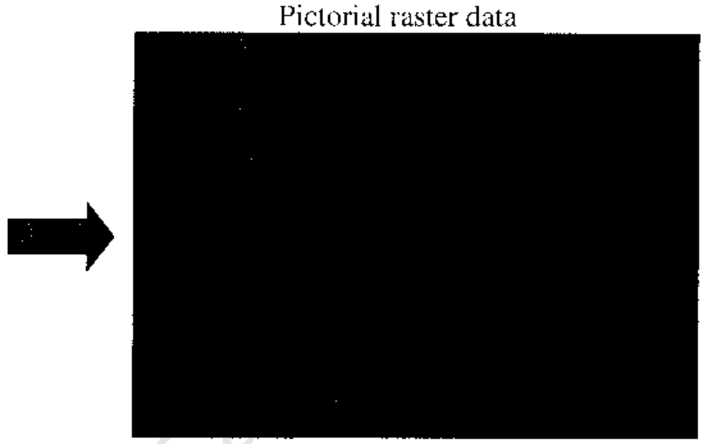


Figure 2.1 (b): Applying a colour scheme to the raster data

2.2 Metadata Standards for geographic datasets

The management of raster datasets usually involves documentation of metadata about the raster datasets. In the (spatial) data management field, metadata is defined as information about the content, quality, condition, and other characteristics of data. Metadata are often called data about data or information about information [30]. The primary benefit of metadata is in its support to the following tasks - data discovery, data access, data transfer, data analysis and data use [31]. Metadata elements describe characteristics that include custodian, description of the data (abstract, purpose, etc), geographical extents of the data, currency of the data, storage format, data quality, and contact information. In this context, metadata are products in their own right and are an important component of any Geographical Information System (GIS) or spatial data management system [37].

Examples of metadata that needs to be documented about raster datasets includes the annotations shown in figure 2.1(a). These include the width of the raster, height of the raster, the cell size (dX, dY), and the real world coordinates of the origin (Xo, Yo).

To allow metadata to be more effectively used by both humans and machines, metadata documents are prepared following some defined metadata standards. A metadata standard is basically a common set of terms and definitions that are presented in a structured format [18].

Metadata standards for geographic datasets include the Content Standards for Digital Geospatial Metadata (CSDGM) from the Federal Geographic Data Committee (FGDC) (FGDC-STD-001-1998) and **ISO** (International Organization for Standardization) 19115 [25]. These standards were meant to provide a structure for describing digital geographic datasets and were intended for, among others, the developers of geographical information systems. These standards define metadata elements, and provide a schema and a common set of metadata terminology, definitions and extension procedures [25]. The standards allow data producers to organize and manage metadata for geographical datasets. They define mandatory and optional metadata elements, to ensure a minimum set of metadata required to serve the full range of metadata applications including data discovery, access and transfer. The **ISO** 19115 was prepared by ISO's technical committee ISO/TC 211 which focuses on geographical information or geomatics.

The FGDC CSDGM set contains the following ten sections [18]:

- Identification Information
- Data Quality Information
- Spatial Data Organization Information
- Spatial Reference Information
- Entity and Attribute Information
- Distribution Information
- Metadata Reference Information
- Citation Information
- Time Period Information
- Contact Information

Appendix **B** shows a more comprehensive list of the FGDC CSDGM elements.

Table 2.1 shows the main metadata sections included in the ISO 19115 schema.

Table 2.1: ISO 19115 - Core metadata for geographic datasets

Dataset title (M) (Citation title)	Spatial representation type (0)
Dataset reference date (M) (Citation date)	Reference system (0)
Dataset responsible party (0) (point Of Contact, Responsible Party)	Distribution format (0) (Distribution Format name and Format version)
Geographic location of the dataset (by 4 coordinates or by geographic identifier) (C) (Extent, Geographic Extent, Geographic Bounding Box or Geographic Description)	Additional extent information for the dataset (vertical and temporal) (0) (extent Temporal Extent, Vertical Extent)
Metadata file identifier (0)	Dataset character set (C)
Metadata standard name (0)	Metadata language (C)
Dataset topic category (M)	Metadata standard version (0)
Spatial resolution of the dataset (0) (Data Identification spatial Resolution, Resolution equivalent Scale or Resolution distance)	On-line resource (0) (Distribution, Digital Transfer Option, Online Resource)
Abstract describing the dataset (M)	Metadata character set (C)
Lineage (0) (Data Quality, Lineage)	Metadata point of contact (M) (contact, Responsible Party)
Dataset language (M)	Metadata date stamp (M)

In table 1 above, M indicates mandatory elements, 0 is for optional and C indicates conditional elements. 'Conditional' is used for one of the three following possibilities:

1. to indicate that a metadata element should be included depending on the value of certain elements
2. to indicate that a metadata element should be included depending on the inclusion of another element
3. to indicate a choice between two or more options, where at least one option should be included and documented [25].

The 'Spatial Data Organization Information' and 'Spatial Reference Information' sections of FGDC CSDGM and the 'Spatial representation information', 'Content information' and `Reference system information' sections of the ISO 19115 allow adequate description of the characteristics of geographic images, owing to the relevant semantics offered in their subsections.

Both the FDGC CSDGM and ISO 19115 define the browse graphic to 'illustrate the dataset'. Although the two standards do not specify what should be shown by the browse graphic, this graphic could be well suited to the provision of a 'quick look' for geographical images before they are acquired by potential users.

The main differences between the two standards are highlighted on the FGDC website (http://www.fgdc.gov/metadata/us-national-profile-iso19115/index_html), including the fact that the ISO 19115 allows for the use of unique identifiers for metadata and can also be used to describe geospatial services.

Another metadata standard in common use is the DCMI Metadata Element Set (DCMES), developed and maintained by the Dublin Core Metadata Initiative [16]. The metadata set comprises the following 15 main elements, each of which is optional and repeatable: *Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, Rights.* [16]

In order to encourage interoperability, DCMI recommends the use of controlled vocabulary for the 'type', 'coverage' , 'date', 'language' and 'format' elements.

While the Dublin Core element set wasn't meant specifically for geographical image datasets, it can be used to provide a basic description of digital geographic (satellite) images. The DCMI Type Vocabulary recognises 'image' as an approved resource type. However, the DCMES doesn't define all the elements which may be required for the developers of geospatial metadata to fully document the geographic images in a standard way. Documentation for geographic images requires additional information like resolution (spatial, radiometric, temporal) and transformation details to convert the image data into real world parameter values.

The inclusion of information for time-series datasets is not explicitly handled by all three standards described above, although some sections of the FGDC CSDGM and ISO 19115 could be adapted to the description of time periods for which the geographic images are relevant. Information on when a time-series starts and the time step of the series are not adequately catered for. An additional issue with the time-series data handling is that a time step of 10-days, usually called a 'dekad' [56], is widely used in the agro-meteorological and early warning communities as a time-step at which reporting is done. This adds a new dimension to time-series data handling which would need to be catered for. The metadata standards outlined above use conventional date systems so they do not naturally cater for this.

The use of raster data usually involves the use of colour tables or legends that at least help users better interpret the data, as shown in figure 2.1(b). Raster image data are therefore usually distributed with accompanying colour tables or information that helps users create legends. Information on legends or colour is not catered for in the metadata schema outlined above.

2.3 Metadata Management Systems

Metadata management systems provide users with means to discover metadata records and learn about the datasets that are relevant to their interests. Metadata management systems also allow providers of resources to input metadata into facilities where they can be discovered. These systems usually hold metadata records that are in a consistent, standardised format.

Metadata management systems present metadata elements or schemas in either textual or graphical form. Geographical metadata schemas typically include spatial and non-spatial components and this has a bearing on the viewing techniques. In most cases, metadata is presented as plain text in text files or HTML documents. Presentation of metadata in graphical form has been limited to sample maps and to maps showing the regions covered by data [1]. The browse graphic defined by the ISO 19115 [25] and FGDC CSDGM [18] as a way of illustrating geographic datasets is an essential component in the visualization of geographic metadata as they provide users with a quick means of gaining an insight and comparing the spatial characteristics of data. A common approach is to use a vector-graphics representation of the spatial extents of geographic datasets possibly superimposed on a vector base map or low-resolution backdrop grid image [31].
Because users of metadata typically study metadata only occasionally and cannot be assumed to be motivated to put effort into learning to interpret complex visualization methods, it is important to ensure that the viewing methods are easy to use [1].

2.4 Geographic Raster Data Storage

The capacity to store and manage huge volumes of image data, usually running into terabytes of data, is an essential requirement of an image management system. Other requirements for the image management storage system include security, backup and recovery capabilities [36].

Three approaches to image data storage and management — file-based, DBMS-based and the hybrid approach are discussed in [31]. The file-based approach is where images are managed as separate files in the file system. In the DBMS-based approach, the images are ingested into the database in the form of binary large objects (BLOBs) or user defined abstract data types. The hybrid approach uses both DBMS storage and file system storage. In this case, the usual scenario is the storage of the attributes of the images in the DBMS while the image data are stored in separate files. The hybrid approach, where links between the image (raster) data and the DBMS stored data are established via object identifiers, is used by many organisations that manage spatial datasets [31]. Entirely file-based approaches are also in widespread use, usually in conjunction with compression techniques to improve on storage efficiency [6].

The inclusion of the satellite images in the database as binary large objects (BLOB) ensures an intact database and that the management of the data benefits from the standard features offered by DBMSs. However, the absence of a standard way of ingesting image data into DBMSs and manipulating it has led to different implementations of the BLOB and abstract data types (ADTs). Examples of these implementations include the GeoRaster object used by Oracle Spatial [34] and SQL Server's Image data type [35]. An improvement in efficiency of retrieving BLOB-based image data from DBMS can be achieved through the use of pyramids and tiles [6, 31].

While there are compelling reasons for the ingestion of images (as BLOBs or ADTs) into databases, there are also compelling reasons to adopt the file-based or hybrid approaches. One of the reasons is that the hybrid and file based approaches allow freedom and flexibility for data managers [31, 58]. Apart from the fact that the storage of binary data is not standardized across different RDBMS, keeping the image data separate in a file server means that they will remain available to other tools that may need to manipulate them [58]. Wu et al [58] also argue that because the images consume gigabytes (and sometimes terabytes) of space, keeping them as separate files result in more efficient I/O access.

2.5 Web-based Raster Data Viewing

Most viewers of geographic images render the raster data at reduced resolution levels. Main reasons for this include copyright and performance considerations [31]. This may also result simply from the need to fit the geographic image in a predefined window size. Visualization of raster data at different resolutions is achieved through various techniques as presented in

[31], including the use of previews or thumbnails, interlaced raster storage, resolution pyramids or image pyramids and spatial multi-resolution encoding.

The Open GIS Consortium (OGC) has developed two implementation specifications that play a crucial role in the presentation of geographical raster data on the web. These are the Web Map Service (WMS) and the Web Coverage Service (WCS) specifications.

The WMS [33] deals with general two dimensional data and dynamically produces maps of geographically referenced data from geographical data. The WMS specification defines three operations — `getCapabilities', `getMap' and `GetFeatureInfo'.

The `getCapabilities' operation returns service-level metadata, which tells the users about what to expect from the service, and how the service should be invoked. This operation returns this information in an XML document that should at least contain information about how to use the `getMap' operation. Below is an extract from an example of the document returned by the `getCanahilitiec ' operation.

```
<GetMap>
    <Format>image/gif</Format>
    <Format>image/png</Format>
    <Format>image/png; mode=24bit</Format>
    <Format>image/jpeg</Format>
    <Format>image/wbmp</Format>
    <Format>image/tiff</Format>
    <DCPType>
      <HTTP>
       <Get><OnlineResource  xmlns:xlink="http://www.w3.org/1999/xlink"
xlink:href="http://dev1.dmsolutions.ca/cgi-bin/mswms_gmap?"/></Get>
        <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
xlink:href="http://dev1.dmsolutions.ca/cgi-bin/mswms_gmap?"/></Post>
      </HTTP>
    </DCPType>
</GetMap>
```

[Source: DM Solutions Group Inc. (DMSG), Canada]

The `getMap' operation returns a map whose geographic and dimensional parameters are well-defined, while the optional `GetFeatureInfo' operation returns information about particular features shown on a map. The `GetFeatureInfo' operation allows the querying of map layers based on supplied coordinates. WMS operations can be invoked using a standard web browser by submitting requests in the form of Uniform Resource Locators (URLs) [33].

The WMS specification standardizes the way in which maps are requested by clients and the way that servers describe their data holdings. For a WMS client to get a map from a WMS compliant server, the following nine mandatory parameters have to be given in the `GetMap' request:

- VERSION (WMS version); REQUEST (request name); LAYERS (comma-separated list of one or more GIS map layers); STYLES (comma-separated list of one rendering style per requested layer); SRS (spatial reference system); BBOX (bounding box lower-left and upper-right in the appropriate units); WIDTH (width in pixels of output map); HEIGHT (height in pixels of output map); FORMAT (format of output map). [57]

Maps produced by the WMS are not the original data and are generally rendered in common graphical formats such as PNG, GIF or JPEG. The Scalable Vector Graphics (SVG) [54] or Web Computer Graphics Metafile (WebCGM) [55] formats, which are both open standards developed by W3C, can also be used to render maps. SVG is a language for describing two-dimensional graphics in XML which accommodates three types of graphic objects — vectors (points, lines and polygons), images and text [54].
The common graphical formats produced by WMS means that the display of the data can be achieved via simple HTML pages in web browsers. More sophisticated viewers can be implemented via the use of JavaScript, VBScript, Java, Java Servlets, Java Applet, Active X Control and browser plug-ins [57].

The OGC Web Coverage Service (WCS) [32] standard offers a set of specifications and operations for retrieving information from coverages, where the term 'coverage' is seen to include geographical layers such as vector (point and polygons) and rasters or grids [7]. The WCS allows compliant server to offer data from coverages in its original form and alongside all available descriptions or metadata instead of just rendering the coverage as a picture, as in the case of the WMS. The presentation of the 'real data' allows complex queries to be performed on these data. A WCS can, for example perform client-side rendering or pass the retrieved data as input into scientific models or other clients [32].
The WCS specification defines the following three operations:

- GetCapabilities - returns an XML document describing the services and descriptions of the data collections from which clients may request coverages
- DescribeCoverage — returns an XML document that fully describes the identified coverages.
- GetCoverage - operation returns a coverage (values or properties of a set of geographic locations), in a specified coverage format [32].

2.6 Web-based GIS Development

2.6.1 Web map servers

"The basic functions of a map server are visualization, navigation and query" [22]. Map servers can be classified based on functionality into the following three categories [22]:

- **static map server —** previously prepared maps (prepared by other tools, likely desktop applications) are offered to the user
- **visualization map server —** a new map is prepared on each request from the user; no GIS functionality offered but emphasis is on cartographic visualisation of the spatial data
- **interactive map server —** a new map created on each request from the user; user can make changes to the map by altering some parameters; attribute and/or spatial queries are offered by the server

Some of the common map server development tools are summarised below.

MapServer is an open source development environment for developing spatially enabled web applications. The MapServer C API supports the viewing of a wide variety of vector and raster formats using the OGR Simple Features and Geospatial Data Abstraction (GDAL) libraries. It also provides support for feature selection by item/value, point, and area. MapServer includes Mapscript that allows popular several scripting languages, including **PHP,** Perl, Python and Java, to access the MapServer C API. MapServer is a cross-platform development environment that runs within Linux, Windows, Mac OS X and Solaris. MapServer features some OGC services, including WMS and WCS. [8]

Deegree (http://deegree.sourceforge.net/) is a Java-based framework for the implementation of local and web based GIS applications. Deegree is available as open source under the terms of the GNU Lesser General Public License (GPL). Deegree offers a Web Coverage Service (WCS) that is able to read coverages from different storage formats and deliver them to any client that is able to perform a compliant HTTP GET or POST request. Deegree supports several raster data formats including GeoTiff, PNG and JPEG. In addition to WCS, Deegree features other OGC specifications, including Web Map Service (WMS). [26]

GeoServer (http://www.geoserver.org) is an open source server that allows the publishing of data as maps or images using the OGC WMS, and vector data using the Web Feature Service (WFS). GeoServer is a GeoTools implementation that is provided under the GNU General Public License. GeoTools is a Java-based library and API for the development of geographical applications (http://www.geotools.org/) and is provided under the GNU Lesser General Public License. GeoTools is a modular open source toolkit that implements OGC specifications for GIS applications.

2.6.2 Web map clients

Open source web map clients that feature OGC interoperability specifications and can handle raster data include **MapBuilder** (http://mapbuilder.sourceforge.net/), **Ka-Map** (http://ka-map.maptools.org/) and **Chameleon** (http://www.maptools.org/chameleon). These clients are highly configurable and can be customised to suit other requirements.

2.6.3 Web-based GIS Development Approaches

The majority of web-based GIS systems follow the client-server approach, consistent with the modus operandi of the Internet [42]. Some 'load balancing' between the server and the client can be achieved by developing client-side GIS applications, resulting in so-called 'thick clients'.

Three general approaches for web-based GIS are listed in [42] as follows:

- "Connectivity-first" approach where geo-processing functionality is handled by web browsers, also known as 'browser-based GIS'
- "Computing-first" augments stand alone or desktop GIS's with netware features, also known as 'internet-enabled GIS'
- "Component-first" approach is whereby GIS functionality is implemented from scratch using a procedural language, such as Java. [42]

The Java Abstract Windowing Toolkit (AWT) offers APIs which can play a crucial role in the development of rich raster data visualisation interfaces. Three of these are listed below.

- **Java 2D API** enables the creation of advanced graphics libraries as well as the creation of image and graphic file read/write filters [45].
- **Java Image 1/0 API -** provides a pluggable architecture for working with images stored in files and accessed across the network [46].
- **Java Advanced Imaging (JAI) API —** allows high performance image processing functionality to be incorporated into Java applets and applications; offers some performance-oriented features including image tiling and deferred execution, whereby pixel information is processed only when needed or just-in-time, avoiding unnecessary

imaging computations. The JAI API interoperates with the Java 2D API to mix overlay graphics data with images. [47]

2.6.4 Techniques for efficient web-based map viewing

As mentioned earlier, the maps generated in the visualisation process are usually rendered in the web browser in common pictorial formats such as JPEG, GIF and PNG. While these are usually very manageable sizes, users with low bandwidth witness slow responses when viewing is done over the web, which has limited the use of Web GIS. To alleviate these problems, [20] discussed the use of the Tecgraf Web Format (TWF) for displaying, browsing and printing of maps. TWF is a file format designed to represent a graphic that contains lines, regions, text and images. TWF aims to allow graphics to be coded into small files without converting vector source maps into raster images. This format allows text, lines, curves and regions to retain their scalable nature and allows them to be directly selected by the user in the browser screen [20].

Web-based GIS application development can benefit from the use of new approaches such as AJAX (Asynchronous JavaScript and XML). AJAX [19] is a web development technique that combines HTML, Cascading Style Sheets (CSS), the Document Object Model (DOM), JavaScript and XML. Google Maps use AJAX to enhance performance efficiency in the visualisation of maps [19].

Other techniques that have been used to allow more efficient display of maps in low bandwidth environments include image tiling. Image tiling helps in the sense that only the relevant portions of an image are displayed, without loading the entire image into memory [28]. This reduces processing and bandwidth requirements and allows users to quickly pan from one section of the image to the other.

2.7 Functionality required in food security early warning

While the accumulation of satellite datasets over time creates data management challenges, it allows those in environment management and early warning an opportunity to compare current trends with historical ones. Also the timeliness and consistency of remote sensing data has compelled early warning practitioners to use remote sensing data extensively in their work. Since remote sensing data is usually presented in digital raster form, most early warning systems require an ability to read and manipulate raster datasets. A common approach in early warning analysis is the comparison of a given period of the current year with the same period of the past year or with mean (or average) conditions for the period, which allows a reasonable estimation of the quality of the current year [24]. Software tools available to early warning practitioners need to be able to perform these and other analyses on raster datasets. The WinDisp software manual [10] lists and explains some of the functionality required in an early warning system that relies on the processing on time-series satellite images. The WinDisp package is a software package commonly used in the early warning community to process raster datasets. It is a desktop public domain software package for the display and analysis of satellite images, maps and associated databases, with an emphasis on early warning for food security, and was originally developed for the FAO's Global Information and Early Warning System [10]. The functionality offered by this package includes:

- comparison of multiple images
- extraction of statistics for graphing trends from a number of satellite images, such as during the crop growing season, for comparison with other years
- computation of new images from a series of images.

While the WinDisp package offers some of the required functionality for food security early warning, it does not offer convenient image discovery tools and users may go through lengthy steps to achieve desired results. The WinDisp software offers some limited automation options, where repetitive actions are handled through the use of simple iterative loops. The package is also unable to make its functionality available on the web and this limits its use.

2.9 Related Work

2.9.1 Metadata management systems

The **GeoNetwork Opensource** is a metadata management system developed by the United Nations' Food and Agriculture Organization (FAO) to facilitate data sharing and exchange between organizations [59, 51]. This metadata portal allows users with necessary privileges to enter metadata about their geographic data or other digital resources following the ISO 19115 standard. Templates for raster and vector datasets are made available to users. Users can also prepare and save their own templates in the system to facilitate faster entry of metadata records for other similar datasets. Appendix A shows the template offered by GeoNetwork to guide users in the provision of metadata for raster data following the ISO 19115 standard.

The GeoNetwork raster metadata template shown in Appendix A does not explicitly force the metadata creator to include a formula (usually linear $y = mx + c$) to transform image data (typically 0-255 for 8-bit images) into values representative of the parameter being measured (say 0 to 0.8 for vegetation index). The ISO 19115 features two elements 'scale factor' and `offset' in the 'Range dimension information' section [25] for this, which appear to have been overlooked by the developers of the GeoNetwork Metadata Catalog, as it doesn't feature in the template shown in Appendix A.

The GeoNetwork portal features a text search facility (figure 2.2) that allows users to discover available records via the use of keywords that are entered into the system by metadata providers.

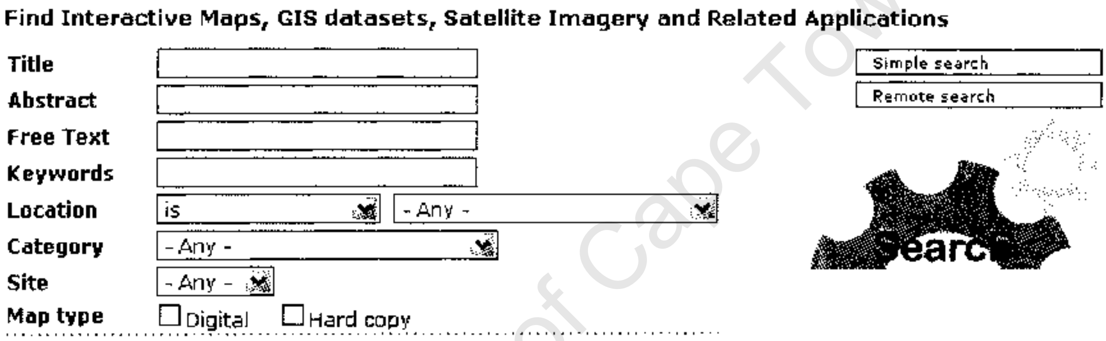


Figure 2.2 GeoNetwork advanced search facility. Source: www.sadc.int/geonetwork

The GeoNetwork portal can optionally be linked with an open source interactive map viewer, Intermap, to facilitate viewing of map or image datasets that are available in the system. Intermap is a generic interactive map viewer that supports the OGC Web Map Service (WMS) standard and ESRI ArcIMS web map service [51].

While the GeoNetwork portal does support the management of spatial raster datasets, it does not provide functions aimed at handling time-series raster datasets. Relationships between datasets are managed by the use of keywords but there are no specific functions to deal with relationships between datasets based on their time period, spatial extent or thematic content.

To handle the 2D nature of image data, the RasDaMan array DBMS [4] features an array based query language, rasQL. RasQl extends SQL92 query language with operators to handle multi-dimensional discrete data.

2.9.2 Raster Data Management Systems

The Global Information and Early Warning System (GIEWS) Workstation [17] is a web-based geographic information system for managing remote sensing, geographic datasets and textual data at global and country levels. This database system was developed by the United Nations' Food and Agriculture Organization (FAO) for handling food security early warning information. The system is developed using Java and has a MySQL database backend. Remote sensing datasets are stored in tables as BLOBs in the database backend when a user enters them into the system. The Java-based application supports several image formats including (ESRI BIL, WinDisp IDA, GeoTIFF, etc). The system features a metadata management system that can allow users to fetch metadata records based on supplied keywords. A map viewer is featured which allows user to visualize display datasets that are included in the database. Geographical map layers can be overlaid in this viewer, while new map layers can also be added.

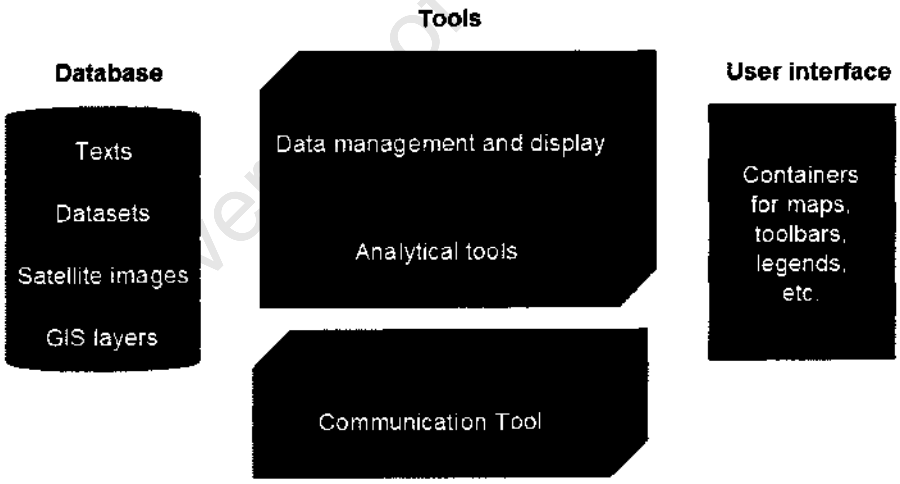


Figure 2.3: GIEWS Workstation architecture [17]

The analytical tools offered by the GIEWS Workstation are not adequate to handle the main routine functions required of an early warning system. Tools are required to assist with preparation of maps for bulletins and ad hoc reports. Functionality to extract statistics from geographic images and to handle image time-series functions, such as averaging and preparation of new graphics on the fly, are not included in the system. Users are also unable to do pixel-based queries on the layers displayed in the viewer.

GrIdS [31] is a DBMS-based raster data management system that offers various functions including raster data insertion and extraction, data mosaic creation, raster data tiling, geo-referencing and visualisation. The system does not however address time-series analysis of raster data.

[6] presents a raster-based database application, RasGEO, which provides interactive navigation on continuous raster datasets as well as ad-hoc generation of some compound raster products. The application is developed as a servlet-based Web application which runs on top of raster data management system known as the RasDaMan multidimensional DBMS [4]. While the RasGEO and RasDaMan combination offers efficient means of storing and retrieving raster datasets, it does not provide adequate functions aimed at managing time-series raster datasets and analysing them to produce results that appeal to the agriculture and environmental monitoring users.

A web-based geographical visualization tool based on OGC specifications, including WMS and WFS, for viewing information related to earthquake occurrence is discussed in [40]. The tool handles grid raster data and allows viewing of several geographic data layers but does not offer further GIS functionality.

Chapter 3 — System Architecture

This chapter outlines design considerations for a time-series raster data management system and then presents a proposed system architecture, along with the decisions that were taken in the development of this proposal.

3.1 Introduction

Remote sensing satellites provide streams of raster datasets used for various environmental applications. Many organizations collect these raster datasets on a continuous basis, with each raster file received becoming part of a large pool of raster datasets in their archives. A typical scenario in these organisations is one in which there are data managers and data users who face different challenges with regard to the handling of raster datasets.

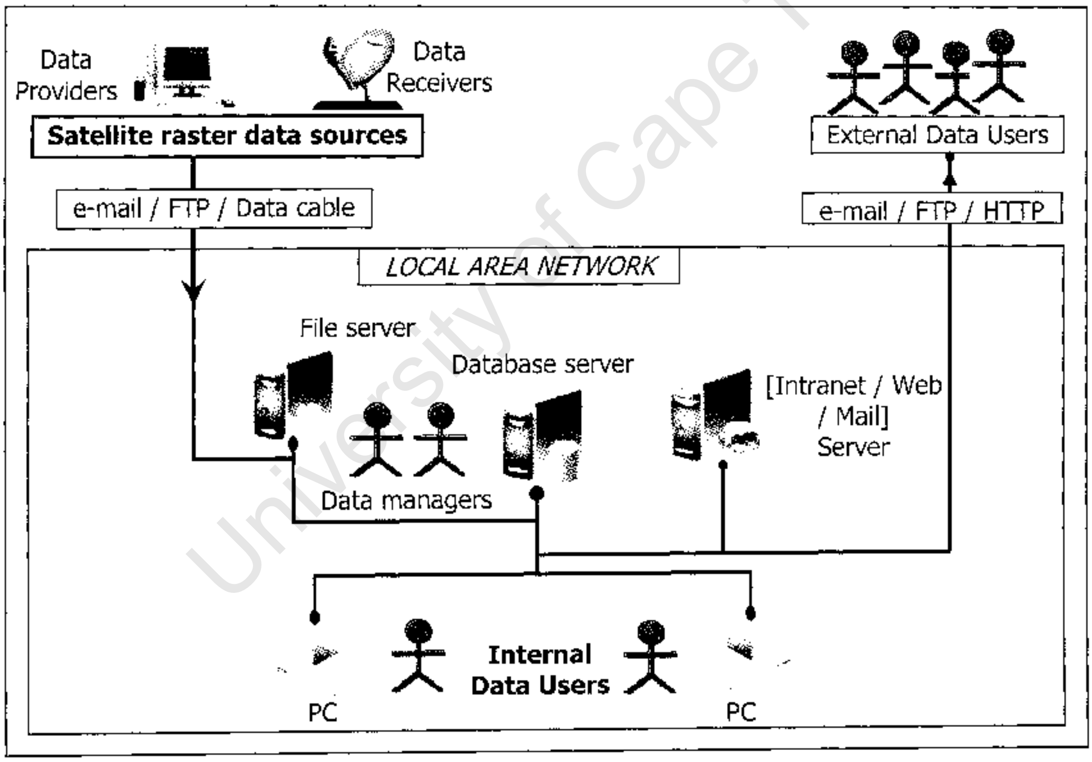


Figure 3.1: Typical scenario for organizations receiving and managing raster datasets

The following sections summarise the typical needs for both the data managers and data users, which influence the design of a possible solution for time-series raster data management.

3.2 Typical Needs for Data Managers

3.2.1 Data formatting

The raster datasets are often received in numerous formats depending on the data providers. This means that data managers have a responsibility to reformat these datasets to suit the demands of their organization before storing the datasets. For this purpose, the data managers need access to a set of data conversion tools.

3.2.2 Data storage

Data storage is often the biggest challenge for data managers, considering the huge volumes of typical raster data streams, both in terms of numbers and file size. A raster data management system must have the capacity to store and manage terabytes of data because of the frequent data acquisitions [36] and the need to maintain historical raster datasets. Data managers require a means to reliably store the raster datasets, and at the same time make them readily available for the users. The storage approaches are usually a choice between file-based, DBMS-based and the hybrid. The advantages and disadvantages of each approach are discussed in [31]. The decision of the data storage option is often largely influenced by available tools that will be used to access the datasets.

3.2.3 Data documentation

In order to promote the easy discovery of data held by the organisation, the data managers need to provide metadata on all the data collections that are held [31, 36]. Because of the volumes of the data involved, the mechanism for the creation of the metadata need to be user friendly and efficient to allow the data managers to easily perform this task. The documentation provided by the data managers informs the users where the data resides and also helps them understand and determine the appropriate use of the datasets. The documentation also needs to include information on how to contact the data provider to ask questions or suggest improvements. For file based or hybrid storage approaches, the file naming conventions used on the files need to be adequately documented and presented to the users. Because this documentation needs to support the entire chain of data discovery and further manipulation, it needs to be as complete as possible. This documentation also needs to be provided on a platform where it can be readily accessed by the targeted users.

3.2.4 Data distribution

For organizations that have a duty to further disseminate the raster datasets, the data managers need to maintain contact details for the external data recipients and the details of the formats in which they receive the raster datasets. They also need to keep records of how to get the raster datasets out to their recipients.

3.2.5 Raster data manipulation

Raster data managers need various tools to manipulate raster datasets as part of their regular routines [31, 36]. They need efficient tools which minimise the manual component in order to save time and also reduce errors in the processing. In anticipation of some raster data analyses which require some reference raster datasets, data managers need tools to prepare reference raster datasets like time-series averages, and other statistical references including median, maximums and minimums.

3.3 Typical Needs For Data Users

3.3.1 Data discovery mechanism

The ever increasing size of raster data archives means that users require efficient and effective means for sharing and retrieving datasets that are available in archives or databases held by their department [36]. In a typical scenario, users rely on standalone desktop systems which do little to assist them with the task of locating files that they require. Users generally require good prior knowledge of the locations of the files that they need before the can use them for any analysis.

3.3.2 Access to data documentation

Data users need platforms to easily discover documentation provided by data managers or providers on how to use the raster datasets and how to get help in using the datasets [30, 31, 36, 51]. This could be addressed by the provision of documents showing the 'frequently asked questions' for each raster dataset and answers to those questions.

3.3.3 Raster data analysis tools

Section 2.8 lists some image analysis functionality relevant to a department that uses time-series raster datasets for agricultural monitoring in the context of food security.

Some of these analyses are applicable to a wider community that handles time-series raster datasets. These include the computation of new images from a series of images. The new images could represent the average, sum, median, minimum and maximum of a time-series collection.

Another common need for data users is for tools to extract information from the raster files [6]. Requirements include extracting information for a given point or area of interest, and analysing the images to get summaries for selected time-series collections of raster data. Overlaying the raster data with other geographical data to help with analysis and understanding of the data should also be provided as a service in a raster data management setup [6, 7].

The raster data analysis tasks are quite often hampered by delays in accessing the collections of raster data required to complete the task, which is essentially a data discovery and access problem.

3.4 Proposed System Architecture

3.4.1 Introduction

To address the time-series raster data management needs outlined in the previous sections, a system architecture is proposed here as shown in figure 3.2.

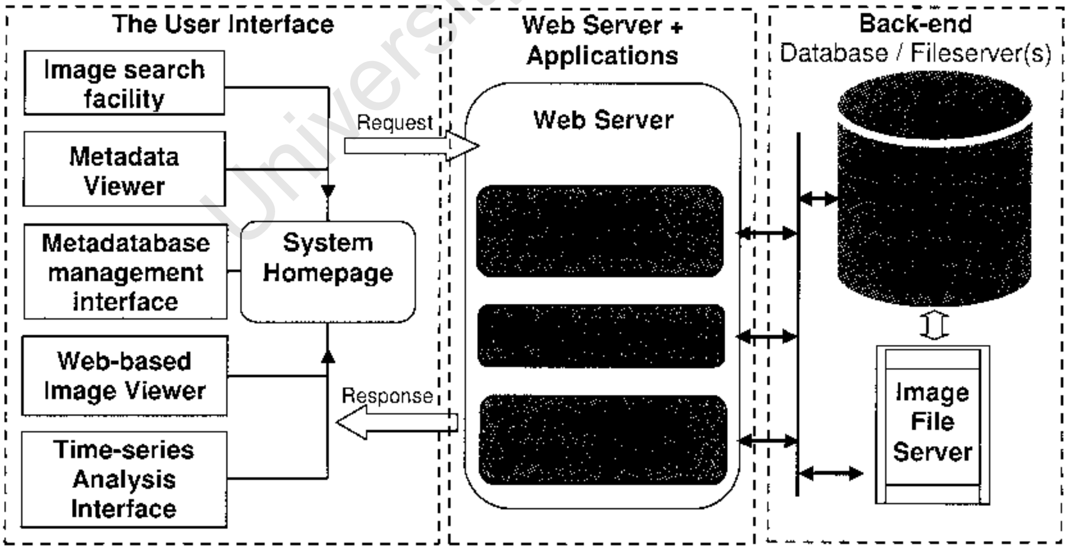


Figure 3.2: System Design

Figure 3.2 shows the proposed system design and the main components proposed. The system architecture is based on a three-tier model [41], which separates the application logic from the client and the back-end. This separation allows the system to be flexible and robust, and also allows the application logic to operate on data from multiple sources [41].

Another motivation for the chosen architecture is that it utilizes only computer hardware resources that are already available to the targeted department.

The following sections describe the components of the proposed solution.

3.4.2 Back-end

The back-end of the proposed system comprises the metadatabase and file servers containing the raster datasets. With regards to the storage of the raster data files and the metadata, the proposal was to keep raster data files on file servers and the metadata in a database. The main reason for this was the fact that the raster data files need to remain available to other software packages which need to access them.

The metadatabase, which is the central part of the raster data documentation and discovery, maps the locations of the raster data files and contains other metadata which helps in understand the usage of the raster data. The full design of the metadatabase is explained in chapter 4.

PostGIS [38] and MySQL Spatial [29], both free and open source DBMS software packages, were considered in the choice of the software for the development of the metadatabase. Postgres/PostGIS open source DBMS enjoys a good relationship with the University of Minnesota's MapServer open source project, which was a strong candidate for the development of the visualisation component, discussed later.

PostGIS was chosen as it offers the functionality required, and also because it offers full support for OGC specifications for spatial DBMS s. The functionality of MySQL is believed to be only suited for simple WMS support and no spatial operations are done on geometry objects [21].

3.4.3 The Application Layer

The middle layer provides web pages for users, performs critical functions of the proposed solution, and retrieves data and metadata from the back-end. These three tasks are handled by separate components that can be modified or replaced independently.

Database Search Engine

The database search engine receives requests carrying the search criteria and connects with the metadatabase to find the matching raster datasets and then prepares a web page listing the results of the search. In the proposed solution, the presentation of the results by the search engine allows the users to get details about each of the raster datasets that are returned by the search.

Web Map Server

The web map server receives requests indicating the raster data sets to be viewed and prepares graphics of these raster datasets ready for display in a web browser. For the development of the web map server, MapServer's Java MapScript API was chosen, as it offers enough tools for the development of a web map server, and is widely used for the development of web mapping applications. The MapServer's Java MapScript API comes with the Geospatial Data Abstraction Library (GDAL), an open source translator library for raster geospatial data that handles many raster data formats, including the Image Display and Analysis (IDA) [10] format, which was one of the main raster data formats targeted.

MapServer can be run as an out-of-the-box CGI program with the HTTP server. For the development of the web map server, the use of MapScript instead of MapServer CGI is preferred because of the opportunity to merge the functionality of MapScript with the functionality of other programming packages. This option provides the possibility of extending the functionality provided by an out-of-the-box MapServer application, for example, to do some extra image manipulation in Java. Use of Java MapScript API also means one can choose which requests are to be entertained by the web map server, apart from the ones already specified by OGC WMS standards, and one can control how the map layers and their legends are drawn and how layer query information will be extracted and presented.

Time-series Image analyser

This component processes image analysis requests from users and prepares a single raster file representing the result of the analysis. It interacts with the database to get the locations of the raster files before reading them from the appropriate file server. Time-series analysis functionality to cater for routine analyses only was considered. The image analysis functions were developed in the form of Java utility classes, mainly because the Java Advanced Imaging (JAI) API [43] offers a wide variety of image processing utilities. Results of the analyses by this component are prepared in the form of raster data files which can be used by the web map server.

3.4.4 The User Interface Layer

To cater for the needs of data managers and users, we proposed the following 6 user interface components.

Metadatabase Management Interface

This interface allows the raster data managers to insert metadata records into the metadatabase. For ease of access this interface is accessible on the Intranet through a web browser.

Image Search Interface

The image search interface allows the users to formulate search requests for the raster data based on keyword, data theme, data type, period of interest, region of interest and cloud cover level of tolerance. This component passes requests to the 'Database Search Engine'.

Metadata Viewer

The metadata viewer presents selected metadata to the users for specified images. This facility allows users to learn more about the records discovered via the image searches. The metadata elements to be presented to the users were determined through consultation with the users.

Browser-based Image Viewer

This component is used to view images returned by the Application Layer. It is technically a WMS client that allows users to pass requests to the web map server. These requests include information on the map layers to be viewed, the geographic extent of the area of interest, the layers to be queried, the position of the point whose information is being queried, and the format of the map or document to be produced. MapBuilder, an open source web mapping client, was chosen for the implementation of the WMS client, mainly because it is easily customisable, and operates well within the Apache Tomcat HTTP service environment. MapBuilder allows a web developer to easily add dynamic maps, including GIS layers and images, from multiple web map servers to their web site. A web developer can customise MapBuilder by building and adding new map widgets to enhance preferred mapping functionality. MapBuilder is a browser-based application that was developed to be fast and interactive and was built using the AJAX [19] development approach.

Time-series Analysis Interface

A component specifically for time-series data was required, to allow the users to specify the period of interest and the type of images that they would like to use, and to save as much time as possible in providing this information. This component passes requests to the 'Time-series analyser'.

System Homepage

The homepage is the means to access all the functionality and features of the system. It is important to provide quick access to functionality for experienced data managers and data users; other ways of navigating can be added as well.

3.4.5 Discussion

This system architecture takes into account the needs of both the data managers and the data users. However, there is a notable bias towards providing more features and functionality for the data users than for the data managers. The assumption was that what most data managers lacked was the means to document the raster datasets in a manner that promotes easy data discovery, and therefore the proposed solution offered the data managers a means to deal with this problem in particular. [31, 36] identify this common need for raster data managers.

The proposed solution (figure 3.2) does not require new hardware in addition to that which is already found in typical raster data management scenarios (figure 3.1).

The fact that the web server also serves as the application server keeps the solution simple and means there is no requirement for additional servers. This could, however, compromise server performance when many requests for raster data or raster analysis are placed simultaneously.

The prototype system does not offer tools to perform some formatting of the raster datasets before they are stored in the back-end. The assumption was that the data managers already have efficient means to perform these tasks and that the bulk of the raster datasets are already in a suitable format. The main data providers that were identified by the users of the system provide raster data that does not require reformatting.

Overall, the proposed system architecture provides enough functionality to have a meaningful impact on the work routines of both data managers and data users. Organizations that deal with raster data management invariably seek to reduce the amount of time spend on manual processing and delivery of data so that they can create some time to focus their efforts on value-added services [36] and the proposed solution should go a long way to allow these organizational to achieve this.

Chapter 4 — Metadatabase Design

This Chapter outlines the design of the metadatabase, including the requirements analysis, the conceptual schema and the physical database design. The development of a data dictionary for the metadatabase is discussed at the end.

4.1 Determination of required metadata

To determine an initial set of metadata items that would be required in the handling of satellite images, documents used in the management of image data at RRSU were inspected.

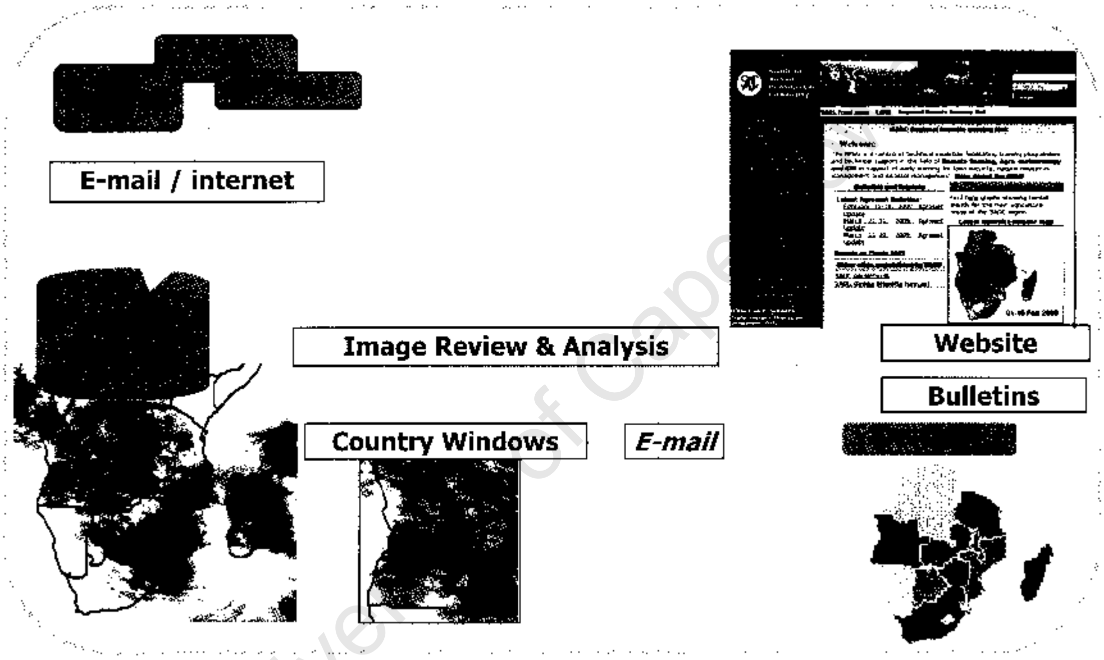


Figure 4.1: RRSU Image Data Handling Activities

The image data handling activities of the RRSU are summarized in figure 4.1 above.
Image data comes to RRSU from several sources in various formats, via either file transfer protocol (FTP) or e-mail. These are stored in the RRSU archive. They are reviewed and analysed before being used in regular reports and bulletins, or chopped into country coverages and distributed to SADC Member States via e-mail. Preparing the image data for the bulletins and publication on the RRSU website involves several stages of processing using various software tools.

The suppliers of the images also supply information that will allow the users to understand and use the images more appropriately. The information supplied usually includes the following:

— the data represented in the images, for example "rainfall"

— the geographic extent of the data, specified as a latitude and longitude range

— image formats, including name of the format used and number of bits per pixel

— image type information (including upper and lower image values, values representing missing data, information used to convert between image pixel values and the real world data represented by the pixel values)

— coordinate reference system (including geographic projection, reference ellipsoids, standard parallel(s), central meridian)

— geo-referencing information (information required to properly position the images so that they can be used in conjunction with other geographic datasets).

Most of the information supplied with the images is not provided in a particular metadata standard format and is usually found in "readme" files compiled by the suppliers.

Table 4.1 — Information typically associated with the satellite images

Information	Example
Descriptive	
Title	"Vegetation index image for January 21-31, 2009"
Theme	Vegetation
Purpose of the image	Monitor vegetation condition
Parameter	normalized difference vegetation index (NDVI)
Valid value range for the parameter being measured	0.1-0.8 for NDVI; 0-253mm for rainfall
Time-series information (whether the data is part of a time-series, start and end of time series, time interval of the series)	21 April 1998 to 31 December 2006
Geographic extent (minimum bounding rectangle, bounding polygon for images)	11 - 45 degrees East; 35 degrees South to 6 degrees North
Data quality information (positional accuracy, cloud contamination, etc)	21% cloud cover; 45 missing lines
Time of validity period represented by the parameter / time period of content	11-20 December 2006
Administrative	
Name of file(s)	V08013
Geographic projection	Albers Equal Areas Projection
Spatial resolution of the image	0.1 degrees

Reference ellipsoid	WGS 84, Clarke 1866
Radiometric resolution	8-bit, 16-bit
Image size (width x height)	250 rows x 400 columns
Image format (n bytes, binary, etc)	8bit binary
Number of bands in the image	1
Byte layout	plain binary, binary interleaved (BIL)
Date of acquisition / entry into the system	22 July 2006
File size (for advice on possible processing or download time)	450 Kilobytes
Distribution information — mode of delivery, format, etc	FTP, e-mail
URL or file location for the image	\\workstationl\images\imagel.iml
Legend (colour map or file)	
Compression information (whether the image data is compressed; if yes, utility to use to uncompress, passwords if applicable)	ZIP, RAR
Usage constraints	None
Source and alternative sources of image (where this image could also be obtained from)	www.earlywarning.usgs.gov/adds
Decoder software list (software that can be used to handle the image)	WinDisp, ESRI ArcView

4.2 User participation

Forms were prepared listing the metadata information types shown in table 4.1. These forms were handed out to the users in hard copy and explanation was given to the users on how to fill the forms. The 3 users chosen were those who manage the data archives at the RRSU and were chosen because of their experience with metadata handling.

The metadata form had headings as shown below.

	Required for storage	Required for display

Proposed metadata were listed in one column. The other two columns in the form required the users to indicate whether the proposed metadata was required for storage in the metadatabase and whether it would be required for display to the user when the user searches for datasets from the database.

The goal of the project was first explained and proposed metadata and functionality proposed for the geo-image visualisation and query interface was given to the users to help them appreciate some of the metadata that would be required to support the prototype system. Users were thus asked to go through the proposed metadata items and functionality proposed for the geo-image visualisation and query interface.

Users filled in the forms and verbally explained the reasons for some of their responses and they also annotated the forms. The feedback from the users, as gathered from the forms, revealed that some important information elements had been omitted in the initial draft. These include the formulas to convert between image data and parameter data and graphic overviews of the image data. These were added to the revised element set.

The responses from the users highlighted the information they would wish to have presented upon request. At a later stage in the metadatabase design, this information would be used to determine the queries to be made on the metadata DBMS, which would in turn determine the relations and the linkages between them.

Scans of the filled forms from 3 users are shown in Appendix C.

4.3 Metadata Element Set

The metadata element set was revised to reflect views and comments raised by the users. Some additional information was accommodated in the reviewed element set, shown in table 4.2. The two additions included are highlighted.

Table 4.2 Revised metadata information requirements

Proposed metadata element / term	**Metadata Standard**
Descriptive	
Title	ISO 19115, CSDGM, DC
Purpose of the image	ISO 19115, CSDGM, DC
Theme (e.g. vegetation)	ISO 19115, CSDGM, DC
Parameter (normalized difference vegetation index)	ISO 19115, CSDGM, DC
Valid value range for the parameter being measured (e.g. 0.1-0.8 for NDVI / 0-253 for rainfall)	ISO 19115, CSDGM
Time-series information (whether the data is part of a time-series, start and end of time series, time interval of the series)	Partially covered by ISO 19115, CSDGM
Time of validity period represented by the parameter / time period of content	ISO 19115, CSDGM, DC
Geographic extent (minimum bounding rectangle, bounding polygon for images)	ISO 19115, CSDGM, DC
Data quality information (positional accuracy, cloud contamination, etc)	ISO 19115, CSDGM
Browse Graphic	ISO 19115, CSDGM
Administrative	
Name of file(s)	
Geographic projection	Bo 19115, CSDGM
Reference ellipsoid	ISO 19115, CSDGM
Image size (width x height)	ISO 19115, CSDGM
Number of bands in the image	iso 19115, CSDGM

Spatial resolution of the image	ISO 19115, CSDGM
Radiometric resolution [8-bit, 16-bit]	ISO 19115, CSDGM
Image format (n bytes, binary, etc)	ISO 19115 CSDGM
Byte layout (plain binary, **BIL,** BSP)	ISO 19115 CSDGM
File size (for advice on processing or download time)	ISO 19115 CSDGM
Date of acquisition / entry into the system	ISO 19115 CSDGM
URL or file location for the image	ISO 19115 CSDGM
Source and alternative sources of image (where this image could also be obtained from)	ISO 19115, CSDGM
Distribution information — mode of delivery, format, etc	ISO 19115, CSDGM
Legend (colour map or file)	Not covered
Compression information (whether the image data is compressed; if yes, utility to use to uncompress (e.g. WinZip), passwords if applicable)	Partially covered by CSDGM
Decoder software list (software that can be used to handle the image)	Partially covered by ISO 19115
Usage constraints	ISO 19115, CSDGM
Formulas to transformation image data into parameter values	ISO 19115

Table 4.2 shows the revised list of proposed metadata and also shows whether or not these are adequately covered by the metadata standards discussed in Chapter 2 — ISO 19115, FGDC CSDGM and Dublin Core. It can be seen that most of the proposed elements are covered by the ISO 19115 and FDGC CSDGM, which are quite comprehensive metadata schemas. Only a few of the proposed metadata are not covered by the reviewed standards. The few metadata elements not covered by the metadata standards reviewed in Chapter 2 primarily represent time-series information. These include, for example, the start and end of each time-series, the interval of the time-series and any gaps that may exist in the series. The principal time-series interval used at the RRSU, the 'dekad' (10 day period), is also not covered by ISO 19115 codes defined to represent different maintenance frequency intervals. The 'Dataset Series' defined by the ISO 19115 is defined generally as a "collection of datasets sharing the same product specification".

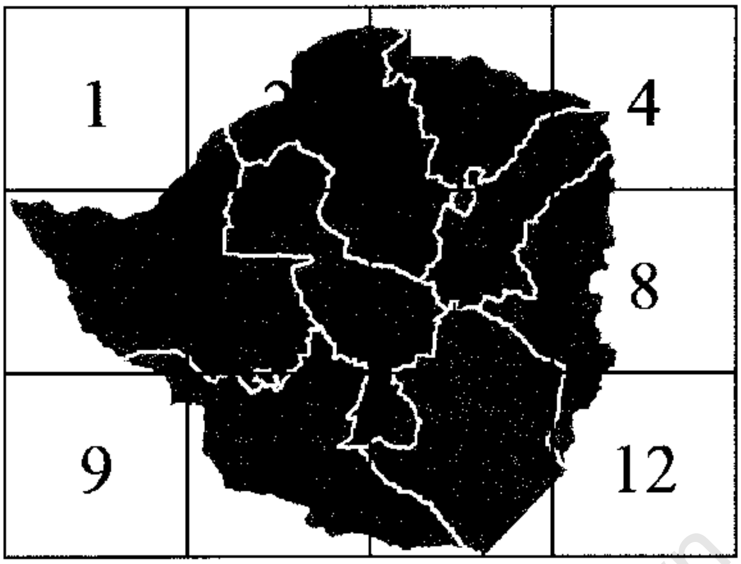


Illustration 4.1: Map series

Illustration 4.1 shows the common understanding of a dataset series as defined by the ISO 19115, in which a series of maps are required to cover one region of interest. This understanding of a dataset series means that the series was not defined in such a manner that it can also be used to adequately represent a time-series dataset. To accommodate a time-series, it was therefore necessary to include metadata elements pertaining to a time-series, including the start, end and time intervals.

Other metadata not covered by the standards relates to the management of colour tables used in the display of images, like the name and location of the colour tables, and their formats.

It should also be noted, that while the ISO 19115 standard defines elements for the lower and upper limit of the data being represented, there are two types of lower and upper limits of interest — one pertaining to the data represented in the image and the other pertaining to the lower and upper limit of the pixel values in the image. For example, we can say that vegetation index ranges from 0.1 to 0.8, but in the image type that represents this data, the image could have a lower limit of 0 and an upper limit of 250. Both types of limits useful are required in order to properly use the images.

Other metadata requirements not adequately handled by the ISO 19115 and the FDGC CSDGM include the compression details and list of software required to decode the image file. For compression, it is required to know the software that would be used to decompress the data, and any password(s) required in the decompression.

The list of software that can be used to decode the images will also help users. Details required here include the operating system on which the decoder software is run and how the decoder should be used. These details are not adequately accommodated by the reviewed metadata standards.

4.4 Metadatabase Schema development

Based on the revised element set described in the previous section, a database schema was prepared, shown in figure 4.2.

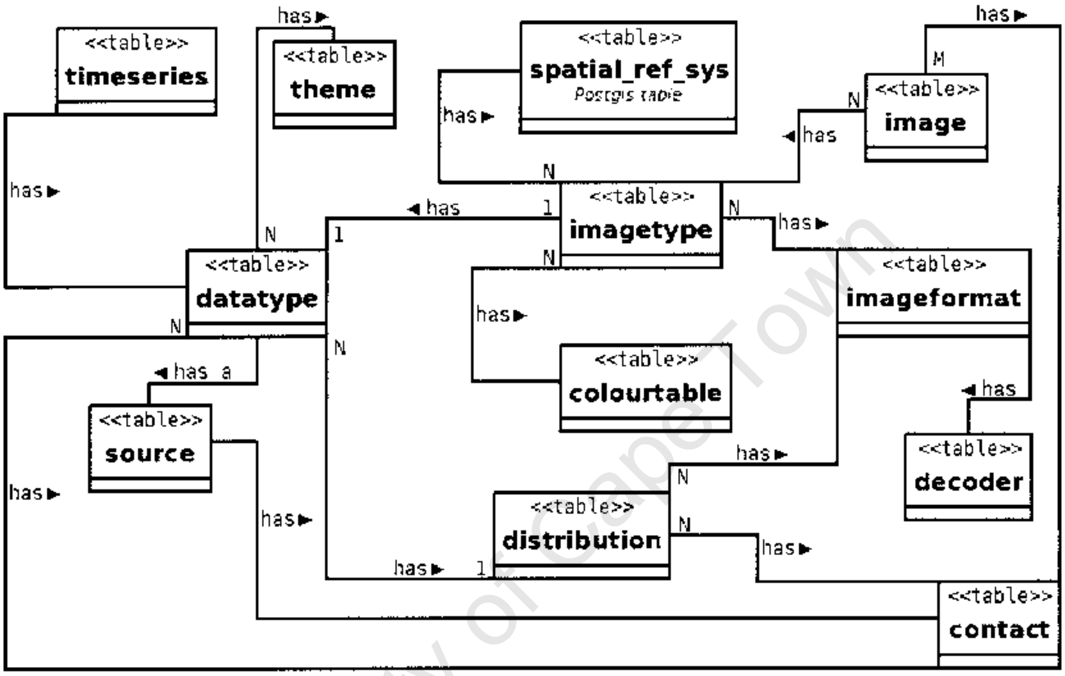


Figure 4.2: UML diagram of the database schema showing all entities.

4.5 Structural business rules for the database schema design

Raster data can be of different **data types,** such as thematic layers including vegetation cover or rainfall totals. Each image **data type** has one or more **sources. A source** (e.g. US Geological Survey Data Centre) can provide more than one **data type.** The image **data type** is distributed by the source via one or more distribution channels (e-mail, FTP). For each image data **source** there is a **contact** person (responsible for distributing the image data). Each image **data type** from a **source** is obtained in a specific **image format. A data type** has a **theme** (e.g. land use, vegetation) and can have a **time-series** (e.g. vegetation data from the NOAA satellite start from 1982 to 2007, at a time interval of 10 days). For each **image type** there is a **colour table** or legend. For each **image type** there is a **spatial reference system** or geographic projection, and an **image format.** For each **image format** there is principal **decoder** (main software used to decode the images).

It is important that metadata also record the people involved with the collection and management of the images. For each **data type** there is a **contact** person (responsible for the data type in the organisation). For each image **metadata** record there is an author or **contact** person.
These relationships are shown by the metadata schema in figure 4.2.

As an example, for a sample image named "v07083", there is one decoder name "WinDisp", image type "SADC NDVI Latlong" with a data type "SADC NDVI" of the theme "vegetation", supplied by source "National Space Science Administration (NASA)" by contact "Ron Smith". Image "v07083" is associated with colour table "ndvie.clr" and has image format "8bit IDA". This image "v07083" is managed by contact "Dorothy at RRSU" and was distributed to SADC countries under distribution scheme "GAC Data".

Figure 4.3 shows the complete metadata schema diagram that was developed for the metadatabase.

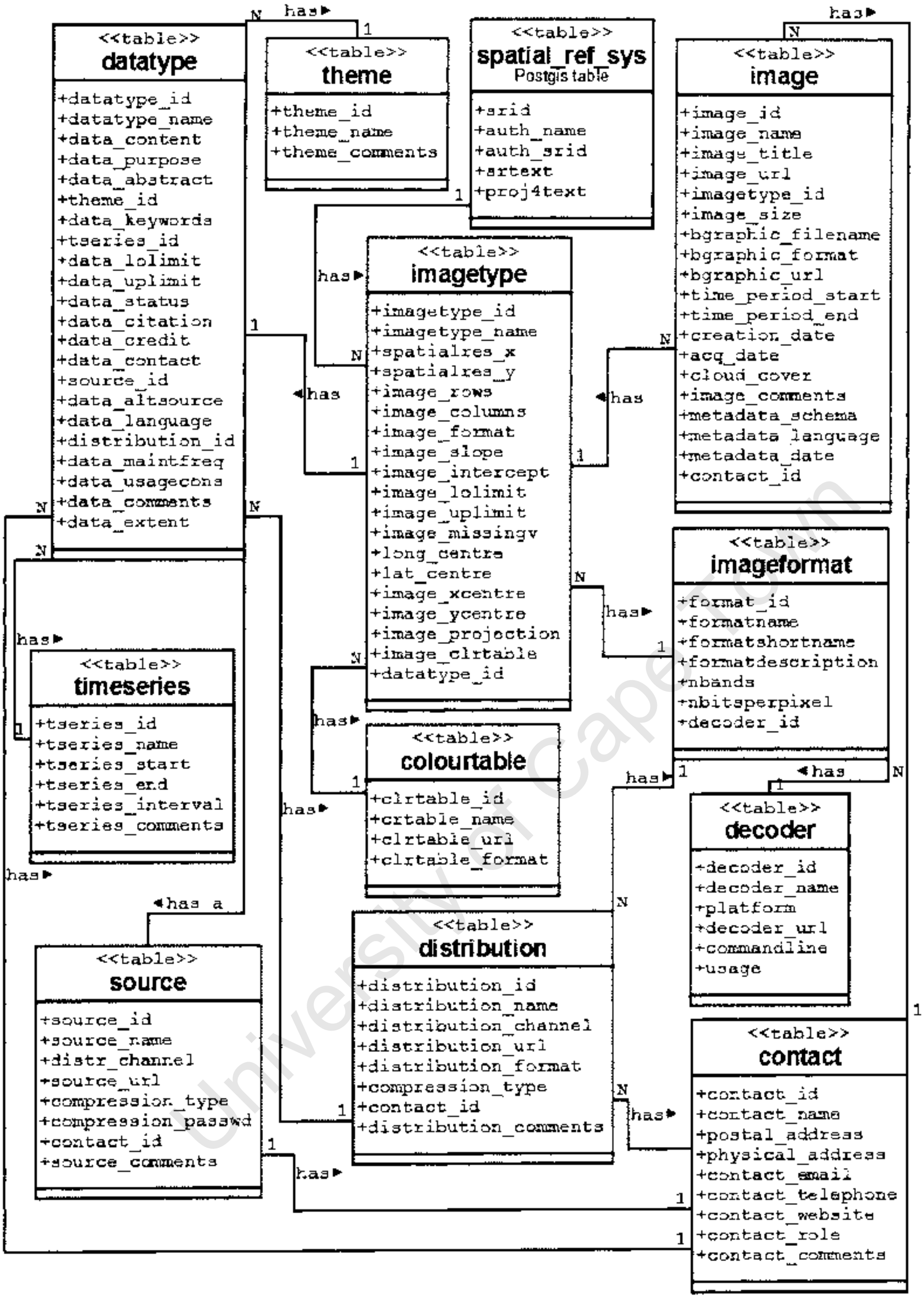


Figure 4.3: UML diagram of the database schema showing all elements.

4.6 Reasons behind the DB schema design

The metadata element set was designed to comply with and extend the chosen standard for geographic metadata, in this case IS019115.

The main tables in the database schema — ***datatype, imagetype*** and ***image*** — were split such that for those images which are part of a time-series, the information is not duplicated every time a new image is acquired. Information which is applicable to all the images in an entire time-series collection is held in the ***datatype*** and the ***imagetype*** tables. While the ***imagetype*** and the ***datatype*** tables could have been joined, they were separated so that the ***datatype*** table deals with the information regarding the world meaning of the data represented in the images (e.g. the parameter being measured, such as vegetation), while the ***imagetype*** table deals with the computer representation of the image itself (e.g. number of bits used to represent a pixel value).

Separating these two tables from the ***image*** table means that only metadata information that is unique to a specific image in a series is captured when the image is acquired. This ***image*** table links to the ***imagetype*** table which in turn links it to the ***datatype*** table so that the entire set of metadata can be obtained about that image when it is required.

It should be noted that the table 'spatial ref sys' is a standard PostGIS table that holds the spatial reference systems that are available [38].

4.7 Preparation of Data Dictionary

A data dictionary listing the metadata elements and their descriptions was prepared and is shown in Appendix **D.** The data dictionary also shows the data types used for the metadata elements and equivalents for IS019115. The possible values for each of the metadata elements are also specified. This dictionary was created so that database users could refer to it when they do not understand some of the attributes used in the various tables. A web page with the data dictionary was prepared and this is accessed through a web link that is provided on each of the database management forms.

Chapter 5 — Prototype Interface Design

This chapter discusses the interface that allows users to interact with the prototype system that was developed for the handling of time-series raster datasets on the Internet. The functionality that the prototype offers is presented at the same time. This prototype is an example of how the system can be presented to end-users; different implementations of the proposed system architecture will differ according to the needs of the end-users they serve.

5.1 The User Interface and Design Considerations

The prototype system was installed on a Linux Server running Apache Tomcat HTTP server. The prototype can also be implemented on a Windows Server with Apache Tomcat with a few minor changes.

The home page of the prototype system is shown below.

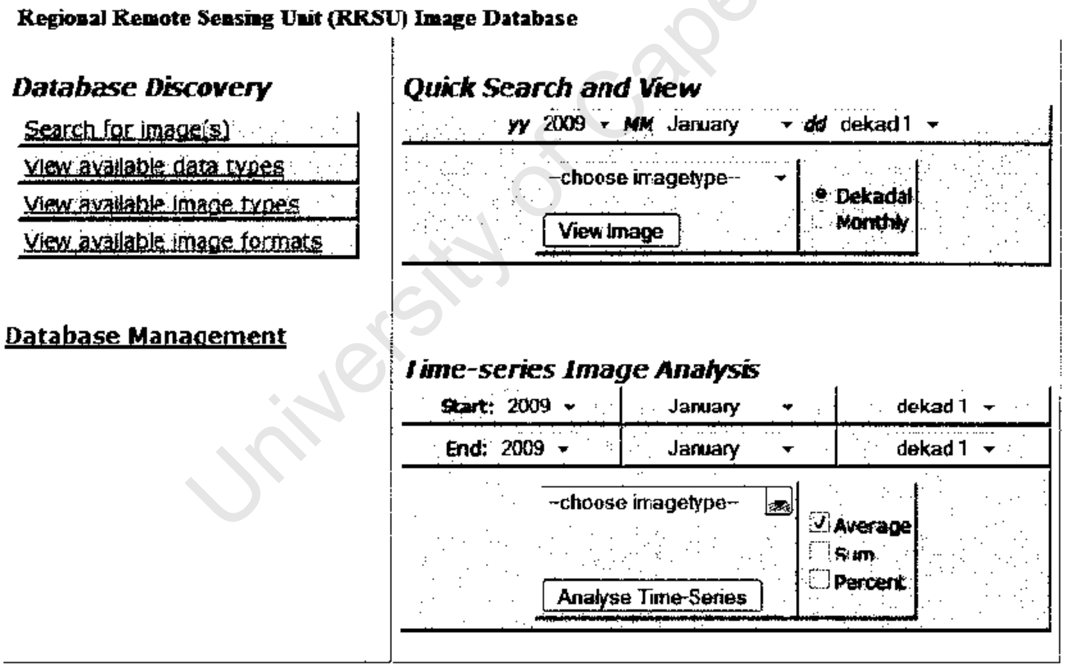


Figure 5.1: Prototype System Home Page

The home page of the prototype provides a text index for quick access to all the important sections of the prototype — the metadatabase management section ("Database Discovery" and "Database Management"), the web map service ("Query Search and View"), and the image analysis functionality ("Time-series Image Analysis"). The home page also offers the users quick links to some useful information like a listing of available data types.

Providing all these features on the homepage also meant that the home page looks quite busy and rather congested. However, fast access to all these important features was considered a priority and the idea was to reduce the number of "clicks" performed to reach any desired item. White space was used to create natural groupings on the screen to separate the different features and reduce the visual complexity of the home page.
The home page and the rest of the user interface were designed to be obvious and self-explanatory. The design ensures that users can easily determine what actions are possible at any moment and constraints in the form of drop-down lists were used where possible to reduce the chances that users will provide information that is inappropriate.

The interface was also designed to have consistency in several ways. For example, consistent fonts and colouring was used on all the database management forms. The colours used are variations of the target department' s corporate colours. The user interface allows the same sequences of actions for most of the services offered, for example, database management interface forms are filled in a consistent manner and submission of information into the database was standard across all the forms.
The following sections will discuss some sections of the interface shown on the home page in figure 5.1.

5.2 Database Management Interface

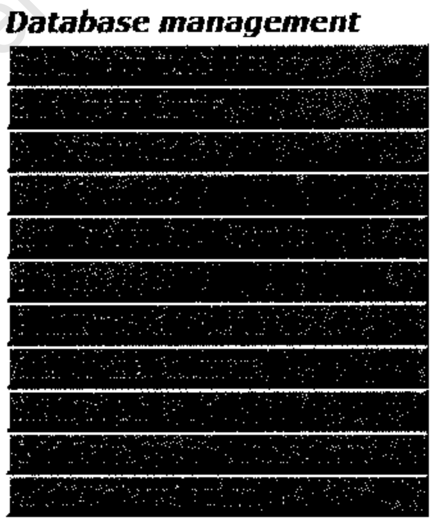


Figure 5.2: Database management interface

Users access the database management interface by choosing the "Database Management" link on the system homepage. The entry of information into the database is facilitated by web forms that can be accessed from the database management section of the interface, shown in Figure 5.2. The common task, adding an image, is the top (first) option; the others are carefully named so that their functionality is self-evident e.g. "Add Colour Table". Links to the data dictionary web page are provided on the database management forms. Appendix E shows the forms provided in the prototype.

5.3 Image Search Interface

5.3.1 Image Search Form

Image discovery was one of the main objectives of the prototype. An HTML link on the metadatabase home-page labelled "Search for image(s)" leads the user to the image search form shown in figure 5.3 below. Available themes, data types and raster dataset sources are provided in drop-down lists. The default region or geographic extent is the area that covers the countries of the SADC region. This can be modified to narrow or expand the search area, by providing the north, south, west and east bounding coordinates. The targeted users have a good geography background and are expected to be able to easily provide coordinates when specifying regions of interest. This allows them to be more precise in the specifications. Some of the users indicated that they would have also wanted a facility that allows them to draw the region of interest on a reference map on the screen using the mouse. Such an interface is generally perceived to be more user-friendly but is less accurate in specifying coordinates of interest. This facility was not implemented due to lack of time but could be pursued in further development of the prototype.

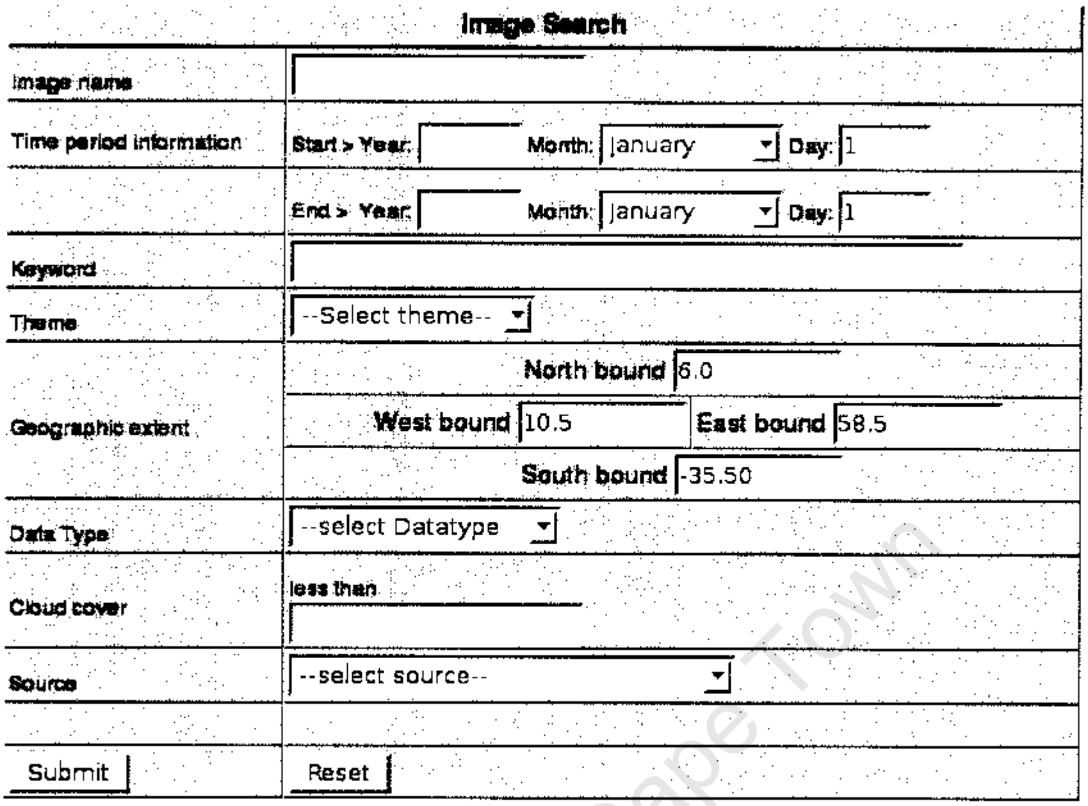


Figure 5.3: Image Search form

5.3.2 Presentation of image search results

On pressing the submit button, the images that meet the criteria are then displayed as concisely yet informatively as possible. Each image is shown as in figure 5.4, giving key metadata along with buttons linking to more detailed views.

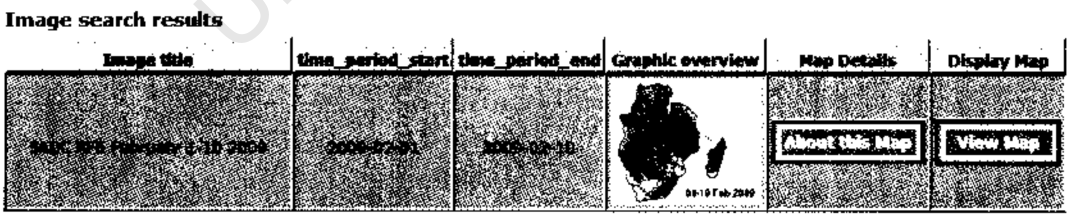


Figure 5.4: Presentation of image search results, with links to further details.

5.3.3 Image metadata viewing

Appendix C shows metadata information items that the users wanted for each image. Further consultation with the users produced a smaller prioritised list of metadata elements. The *"About this Map"* link leads users to a web page that shows this list of selected metadata elements alongside the image thumbnail (browse graphic) as shown in figure 5.5. Links to the next or previous images in the time series collection are provided.

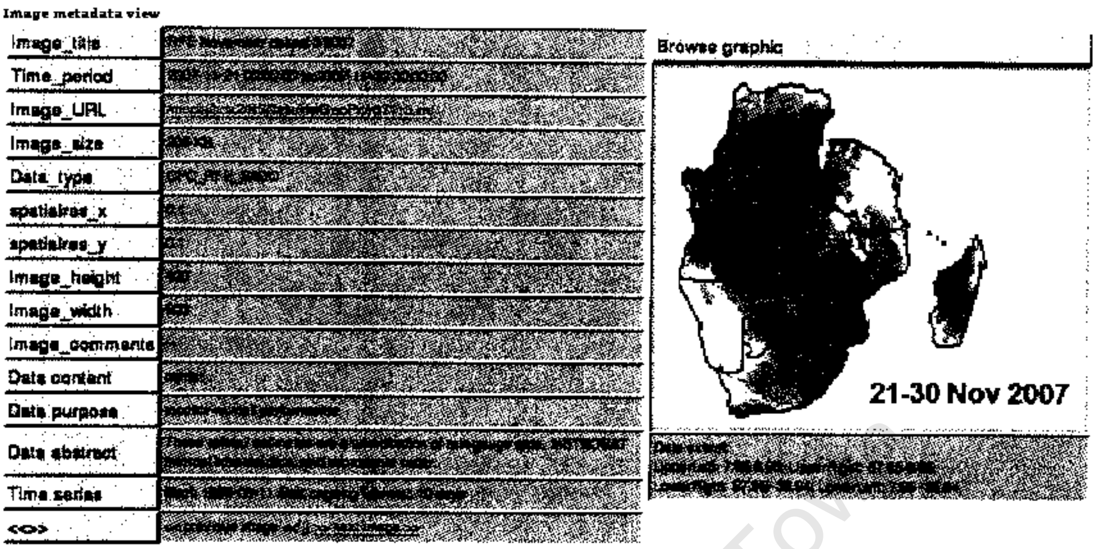


Figure 5.5: Image Metadata viewer

Figure 5.5 also shows a link "View Map" that allows the image to be viewed in the web map service.

5.3.4 Quick search and view

The home page of the prototype system also provides another option for searching for raster data from the metadatabase as shown in figure 5.6. This option only allows searches based on the time period and the data type of the image. The time period is specified as the year, the month and the dekad number of the month. Dekad numbers are 1-3, to represent the periods, 1 to 10, 11 to 20 and 21 to end of month.

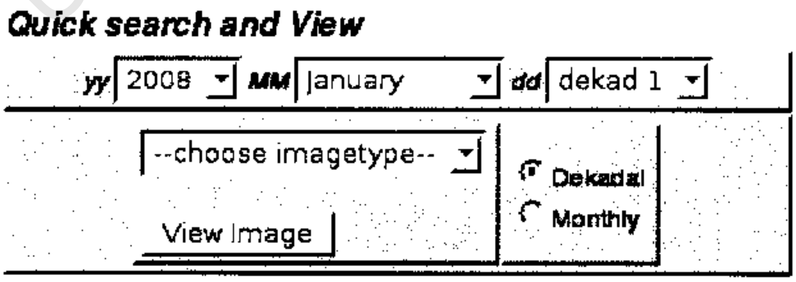


Figure 5.6: Quick image search and view

If a user chooses the "Monthly" option shown in the figure 5, the search will return all the images for the specified month, for all the 3 dekads. The "Dekadal" option returns only the image for the one dekad specified.

5.4 Image Viewing Interface

5.4.1 Image Viewer

The web map service offers an "image viewer" to facilitate the viewing of the images discovered via the metadatabase searches. The image viewer displays the image as shown in figure 5.7 below. A reference map is given in the bottom-right corner of the image viewer with an indication of which region the image data covers.

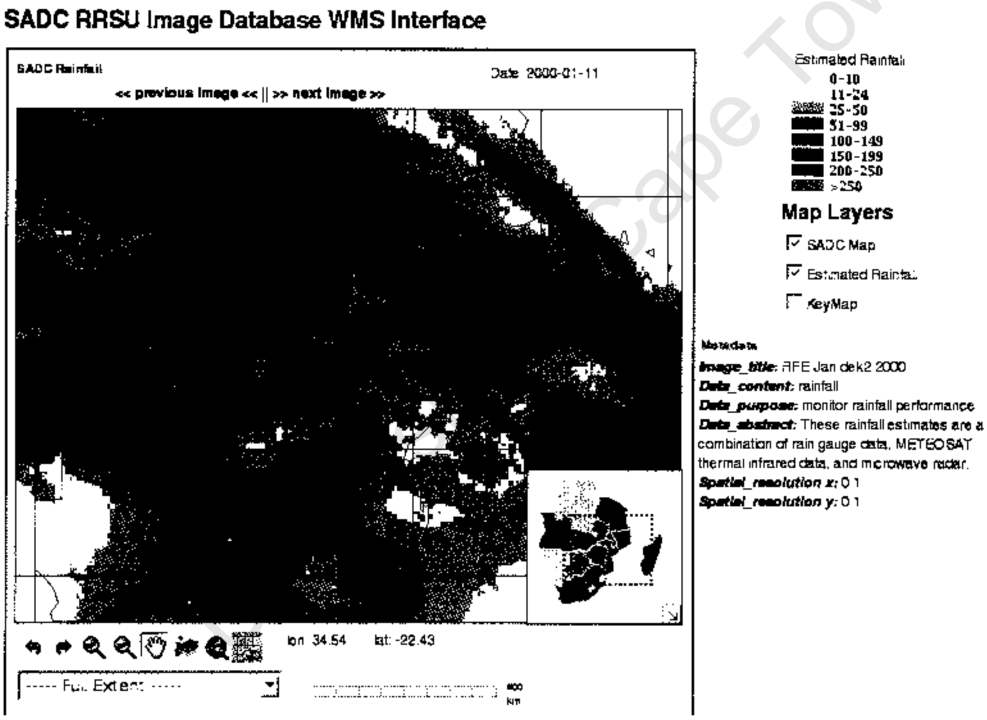
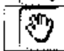
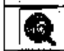


Figure 5.7: Image Viewing interface

5.4.2 Navigation features

The viewer interface offers navigation features — zoom, pan — to allow users to explore the image or raster data. A map overview (insert) is provided to allow the user to see which part of the SADC region they are viewing. Latitude and longitude coordinates are also displayed for the point over which the mouse is hovering. A drop-down list is provided which lists all SADC countries and allows a user to zoom directly to any country of their choice. The buttons below the map are standard buttons for common GIS functions, as detailed in Table 5.1.

Table 5.1: Map Interface navigation features

Navigation Feature	Purpose
	Back to the last geographic extent viewed
	Next geographic extent viewed
	Zoom in
	Zoom out
	Pan
	Get information for position on map
	Zoom to full extent
	Show latitude / longitude grid

5.4.3 Legend

The legend of the image is retrieved and presented as a graphic (either PNG or GIF) on the top right of the viewing interface to help users interpret the colours shown in the image viewer.

5.4.4 Available GIS Layers

Other GIS layers loaded into the viewer are listed just below the image legend. Users can use the check boxes provided to select or de-select the layers depending on which ones they want to view.

5.4.5 Metadata

Some essential metadata records are retrieved from the metadatabase and presented alongside the image, shown on the right in figure 5.7. These records were selected based on consultations with users.

5.4.6 Time-series navigation

The user can view other images which are in the same time-series as the current image by using the links *"<< **previous** <<"* and *">> **next** >>"* as shown in figure 5.7, just above the image being viewed. These links are only activated when the previous or next images in the series are available. The date of the image is displayed on the interface so that the users can keep track of the period the image represents. This date is part of the metadata records retrieved from the metadatabase.

5.4.7 Image pixel-based queries

The user can select the information tool and click on a point of interest on the map. The information for the point, for example vegetation condition or rainfall amount received at the point, is then presented as either XML or HTML (default is HTML).

54 Image Analysis Interface

Image analysis functions are accessed through the home page as shown in figure 5.8 below.

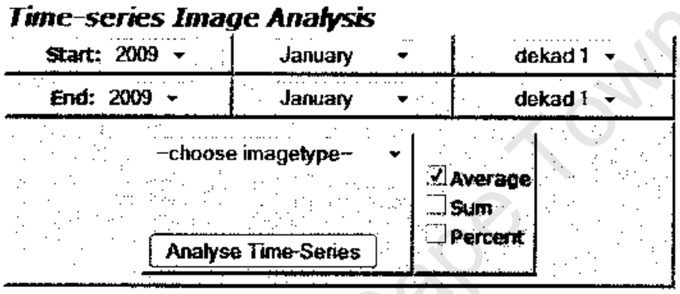


Figure 5.8: Interface for accessing image analysis functions

This allows users to specify the start and end of time-series period of interest and provides a drop-down list of human-readable image type names. Check boxes are provided to specify which type of image analysis the user wants to perform. The currently implemented functions are ***image averaging, image summation*** and calculations of ***percentage of average*** (comparison with average) as these are the most commonly used, based on user interviews and consultation of reports produced. The ***percentage of average*** function provides a comparison between current conditions and reference (average or expected) conditions. The function gives the user an image which is the result of dividing the sum of all the images in the specified time interval by the sum of the images representing the averages of each of the periods in the interval, and multiplying the result by 100. See Figure 6.9 (section 6.3.3) for an illustration of this function.

The *'Analyse Time-Series'* button will initiate the processing and invoke the web map service to display the results as in figure 5.7.

An overview of the prototype system and its interface is shown in figure 5.9.

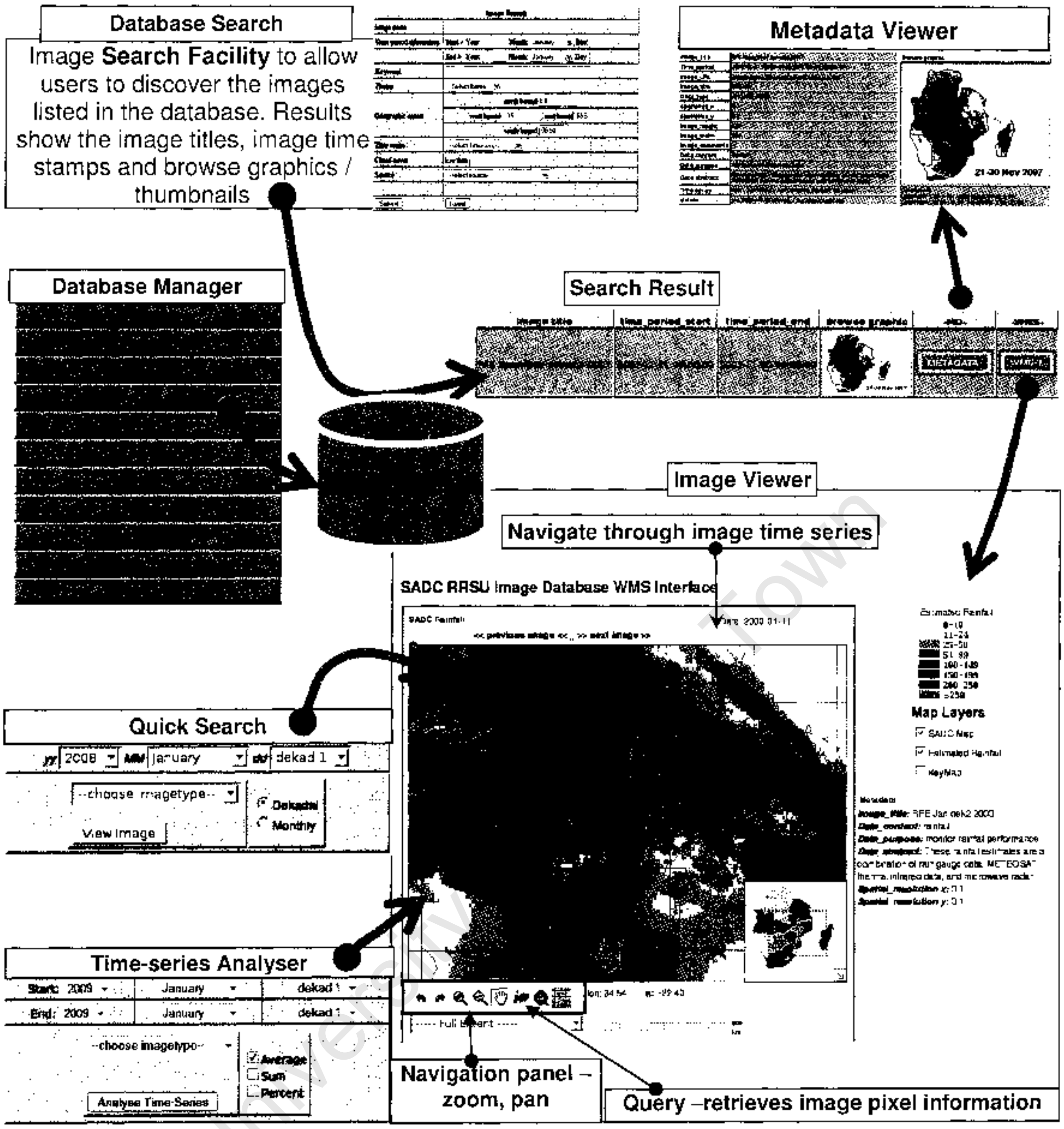


Figure 5.9: Overview of the components of the user interface

Chapter 6 — Prototype System Implementation

This chapter presents the steps that were taken in the implementation of the prototype system for the management of time-series raster datasets. The chapter is presented in three sections — metadatabase implementation, web map service implementation and image analysis functionality implementation.

6.1 Metadatabase Implementation

6.1.1 Database Creation

PostgresSQL (version 8.2) and PostGIS software were installed. The database was created using the following SQL statement:

CREATE DATABASE rrsu_imagedb WITH ENCODING='UTF8' OWNER=postgres TEMPLATE=postgistemplate;

The database 'postgistemplate', an in-built template, is used so that the new database 'rrsu_imagedb' is spatially-enabled. Spatially enabled here means that the database has spatial functions and in-built tables required to handle geometry.
A PostgreSQL database user 'postgis', was created for the day to day transactions on the database. All the tables created in the image database were given 'postgis' as the owner to allow 'postgis' full privileges (updating, viewing, and delete) on the tables.
SQL statements for the creation of the database tables are shown in Appendix F.

6.1.2 Population of Metadatabase

Before the database could be used for the entering of image metadata, some base data had to be collected and entered into the database. The main inputs are listed below.
The population of the database started with the tables which do not make references to other tables. The following tables were thus populated first:

— ***Themes*** **—** The following themes, chosen by users, were inserted into the database: *precipitation, land use/cover, vegetation, agro - ecology, cloud cover, elevation*

— ***Colour tables*** **—** colour legends or schemes used in displaying and analysing the images were identified in the RRSU working papers

— ***Contacts*** **—** this included details of:

  — contacts at the institutions or organizations supplying the images

— persons responsible for image data types within RRSU
— RRSU persons responsible for distributing image data
— users who create metadata records for the image data

— ***Geographic Projections*** – the default spatial references table (spatial_ref sys) that comes with PostGIS was modified to include projection systems used by the RRSU. This was because the spatial references systems in the table were primarily suited to northern hemisphere mapping.
— ***Image formats*** – list of available image formats managed by the RRSU
— ***Decoders*** – names and characteristics of software used to handle images at RRSU
— ***Timeseries*** – information such as start, end dates and time intervals collected from documentation provided by the image data suppliers.

The next set of tables populated made reference to the tables above:

— ***Sources*** - suppliers of the image data (this makes references to the contacts table).
— ***Distribution*** — different schemes for image data distribution from RRSU to the stakeholders in Southern Africa, gathered from RRSU working papers (this table makes reference to the 'contact' and the 'imageformat' table).
— ***Image types*** – populated using documentation provided by suppliers (this makes reference to the 'imageformats', 'spatial_ref sys', 'datatype' and 'colourtable' tables).
— ***Dataset types*** – populated using documentation provided by suppliers of the image data (this table makes reference to the 'contact', 'source', 'theme', 'timeseries' and 'distribution' tables).

The SQL used to load the database were captured as scripts so that the loading process could be repeated easily.

6.1.3 Metadatabase Interface Development

To ease the management of the database and the discovery of image metadata, a visual interface had to be developed. Java Server Pages (JSP) was chosen as the scripting language for handling the interaction between the interface and the database. This choice was mainly influenced by the requirement that the appropriate technology should be usable on all the main operating systems (including Windows and Linux), and that it should be a freely available technology. PHP and JSP were the main candidates for this purpose but Java was preferred as experience suggested that it would be easier to develop JSPs and Servlets because of the availability of Java Development Kits (JDKs) which provide user-friendly environments for developing, compiling and testing source code.

JSP and Java Servlets are powerful utilities used in the development of dynamic web-based interfaces. For connecting to the database, the freely available Java Database Connectivity (JDBC) driver was used.

Apache Tomcat (Apache Software Foundation, 2008) [3] was used as the Java HTTP web server environment for execution of Java code.

6.1.4 Development of web forms

Database entry forms consistent with the proposed metadata schema were prepared mostly as static HTML pages. Some of the web pages had to be developed using JSP since they contained information read from the database.

The web forms used the 'POST' method to post data to the processing JSP or Java Servlets. File upload buttons were employed to allow users to browse for images. However, since the form does not transmit the URL of the file when posting data, hidden text inputs were used to receive the URL of the file from the file upload textbox. JavaScript functions were developed to transfer the URL from the file upload form element to the hidden text input. Figures 6.1, 6.2 and 6.3 below show the three web forms that are used to capture most of the metadata pertaining to the raster datasets.

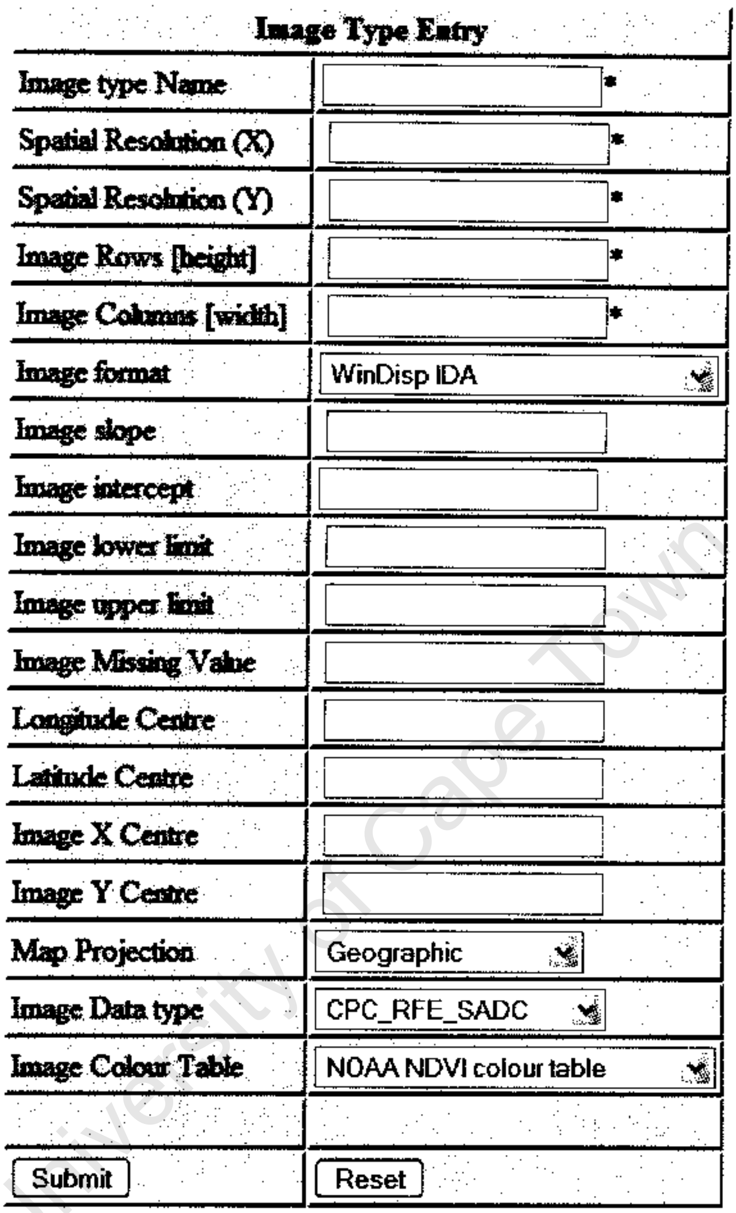


Image Type Entry	
Image type Name	*
Spatial Resolution (X)	*
Spatial Resolution (Y)	*
Image Rows [height]	*
Image Columns [width]	*
Image format	WinDisp IDA
Image slope	
Image intercept	
Image lower limit	
Image upper limit	
Image Missing Value	
Longitude Centre	
Latitude Centre	
Image X Centre	
Image Y Centre	
Map Projection	Geographic
Image Data type	CPC_RFE_SADC
Image Colour Table	NOAA NDVI colour table
Submit	Reset

Figure 6.1 Image type metadata entry form

Data Type Entry	
Data type Name	*
Data content type	*
Data purpose	*
Data abstract	
Data Theme	--choose Theme--
Keywords	
Time series	--choose TimeSeries--
Data Lower limit	
Data Upper limit	
Data extent [polygon]	Upper-left Upper-right Lower-right Lower-left
Data status	Completed
Data citation	
Data credit	
Contact	--choose Contact--
Main source	--choose Source--
Alternative source	--choose Source--
Data language	English
Data distribution scheme	--choose Distribution scheme--
Maintenance frequency	
Usage constraints	
Comments	
Submit	Reset

Figure 6.2 Data type metadata entry form

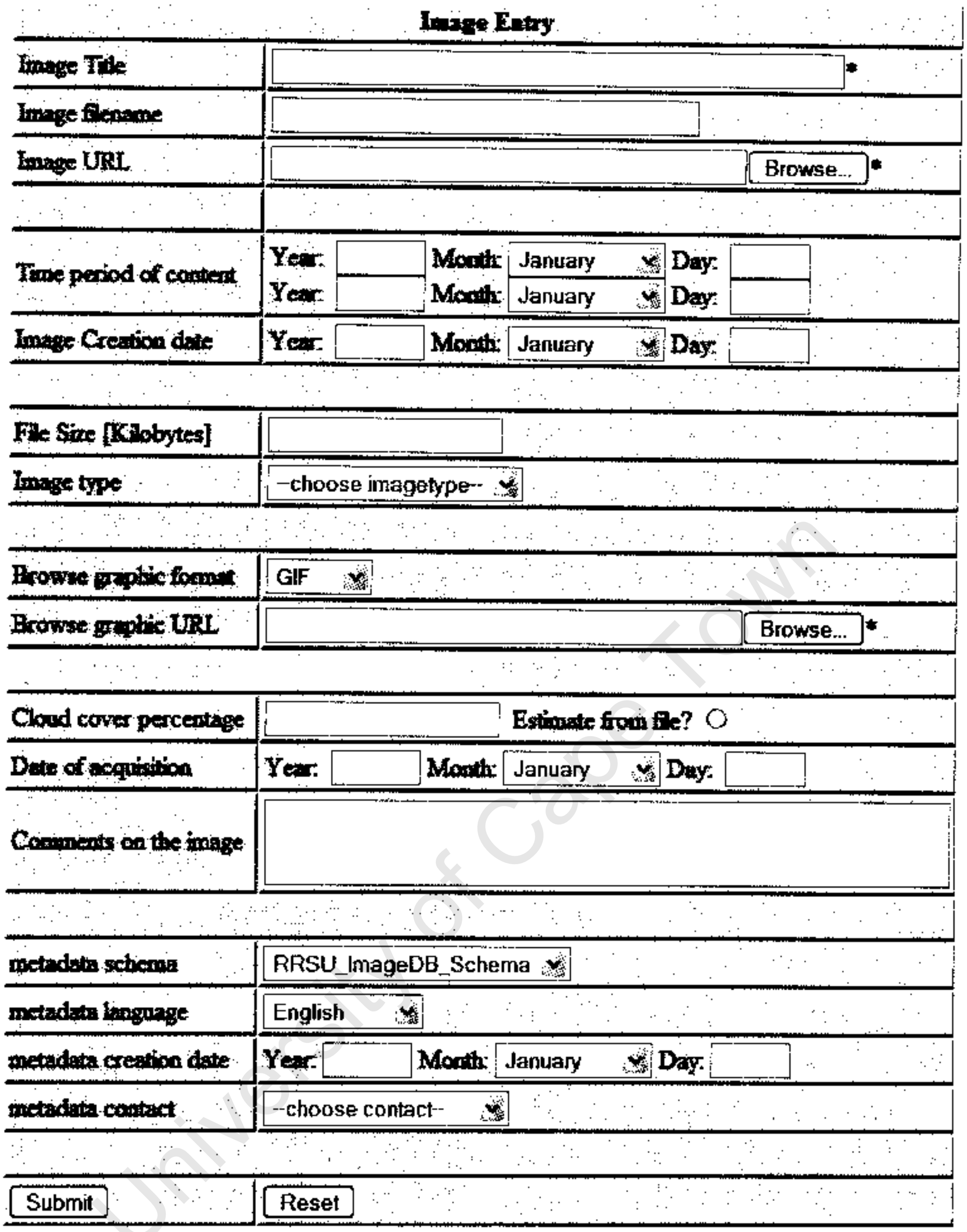


Image Entry	
Image Title	*
Image filename	
Image URL	Browse... *
Time period of content	Year: Month: January Day: Year: Month: January Day:
Image Creation date	Year: Month: January Day:
File Size [Kilobytes]	
Image type	--choose imagetype--
Browse graphic format	GIF
Browse graphic URL	Browse... *
Cloud cover percentage	Estimate from file? ○
Date of acquisition	Year: Month: January Day:
Comments on the image	
metadata schema	RRSU_ImageDB_Schema
metadata language	English
metadata creation date	Year: Month: January Day:
metadata contact	--choose contact--
Submit	Reset

Figure 6.3 Image metadata entry form for time-series images

6.1.5 Development of Java Server Pages and Servlets

Java server pages and Servlets were developed to communicate with the web forms and process data supplied by the user of the database management interface. The information entered in the web forms was retrieved using the ***getParameter()*** method of the ***request*** object.

The JSPs and Servlets were used for executing SQL statements and for processing and displaying the results. The choice between JSP and Java Servlets was primarily influenced by the amount of static HTML on the output web pages. Those that had a large amount of static HTML were done using JSP. Where there was more Java code that static HTML content, Java Servlets were used.

6.1.6 Database Image Searches

A Java Servlet handles the image searches and returns a web page with the results. The form shown in Figure 6.4 was developed to capture the search criteria.

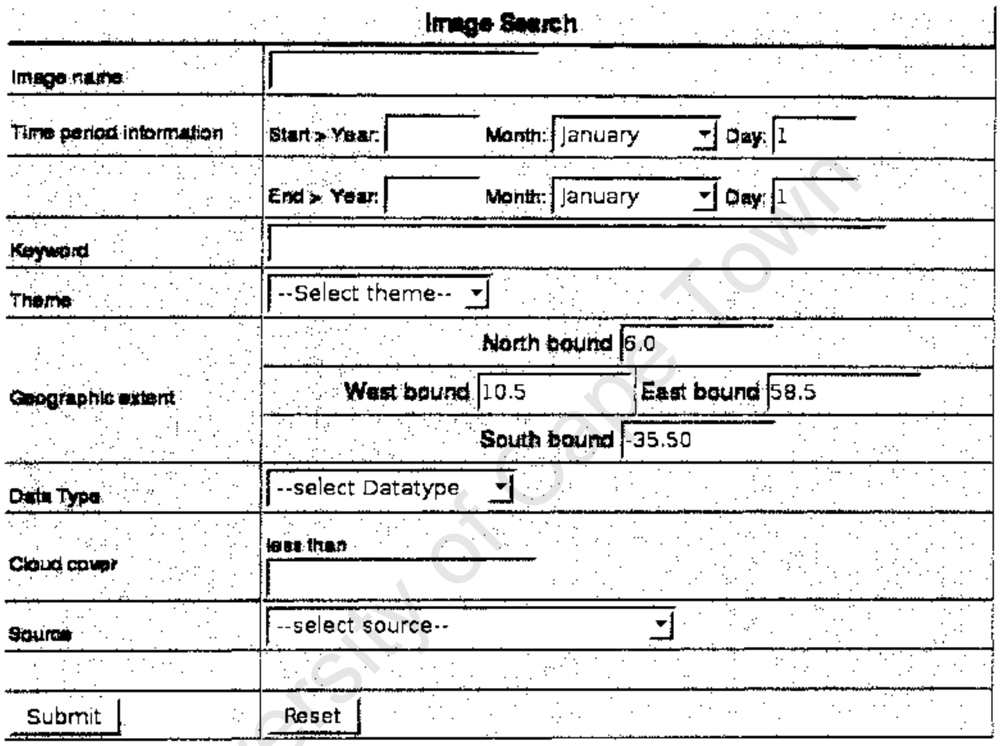


Figure 6.4: Image Search Form

6.2 Web Map Service (WMS) Implementation

The Web Map Service, which allows the viewing of raster datasets and other supporting GIS layers, was implemented using Java MapScript. Java MapScript was used in the form of a dynamically loadable module that makes MapServer's MapScript classes available in a Java environment.

6.2.1 Setting up Java MapScript

To allow Java MapScript to be recognised in the Java Servlets and JSPs, the MapScript Java Archive file (mapscript.jar) was placed in the Apache Tomcat *common/lib* folder so that it is automatically loaded when Apache Tomcat starts.

6.2.2 MapServer Mapfile

The MapServer Mapfile is the basic configuration mechanism for MapServer which allows the user to specify a listing of map layers to be rendered by the WMS and how to render them. The order in which the layers are listed is the order in which they will be drawn, but this order can be changed dynamically using Java MapScript.

One of the most important sections of the Mapfile is the section where the data for each layer is specified, as follows:

```
LAYER
 STATUS ON
 NAME "time_idx"
 TYPE POLYGON
 PROJECTION
   "init=epsg:4326"
 END

 DATA "data_extent from (select image.image_id, datatype.data_extent, image.image_url,
image.time_period_start from datatype, imagetype, image where datatype.datatype_id=2 and
datatype.datatype_id=imagetype.image_datatype and
imagetype.imagetype_id=image.image_type) as sadcgeom using unique image_id using
SRID=5001"

 METADATA
   "wms_title" "TIME INDEX"
   "wms_srs"   "EPSG:4326"
   "wms_extent" "11.0 -35.0 58.5 6.0"
   "wms_timeextent" "2000-01-01/2008-04-21/P1D"
   "wms_timeitem" "time_period_start" #column in postgis table of type timestamp
   "wms_timedefault" "2000-01-01"
 END

 CONNECTION "dbname=rrsu_imagedb user=user1 password=pwd port=5432
host=localhost"
 CONNECTIONTYPE postgis
END
```

The Mapfile also specifies the extent of the map and the default output format for creating the output image graphic.

6.2.3 Coordinate Systems file setup

The Mapfile specifies coordinate reference systems or geographic projections for all listed map layers. The geographic projections are specified as EGSG codes, which are predefined numeric codes associated with coordinate system definitions. For instance, EPSG:4326 is geographic projection using latitude and longitude references based on the WGS84 earth representation model [39]. The WMS protocol uses EPSG codes to describe coordinate systems. EPSG codes are published by the OGP Surveying and Positioning Committee [39]. A list of definitions corresponding to the EPSG codes is found in the EPSG file which has to be located in a convenient known folder [39].

As most of the projection systems specified in the EPSG file are relevant only to the northern hemisphere, it was necessary to edit this file to include some more projection systems that apply to Africa and southern Africa. In the initial population of the database (section 6.1.2), some projection reference systems were added to the 'spatial_ref sys' table. These were also added to the EPSG file to allow them to be used with the WMS.

6.2.4 Web Map Server Development

The Web Map Server (WMS) was developed as a Java Servlet, using the Java MapScript API. The WMS Servlet receives information from the client using the GET method, where the data is sent as part of the URL.

The WMS Servlet receives the parameters shown in table 6.1.

Table 6.1 Information received by Web Map Server

Parameter name	***Explanation / possible values***
VERSION	WMS version, default 1.3.0
REQUEST	One of GetMap, GetFeatureInfo, GetCapabilities, GetLegendGraphic; default is GetMap
WIDTH	width of map to be prepared
HEIGHT	height of map to be prepared
FORMAT	Output image format
CRS / SRS	Map coordinate reference system
LAYERS	'Map layers to be processed
QUERY_LAYERS	Map layers to be queried
INFO_FORMAT	Output format of query information
FEATURE_COUNT	Number of features to be returned by query
X	X coordinate of point to be queried
Y	Y coordinate of point to be queried
STYLES	A predefined style to use in rendering the map layer
BGCOLOR	Background colour
TRANSPARENT	whether output images should have transparency
ELEVATION	The height of the required geographic information
TIME	Timestamps value for temporal data
BBOX	Geographic extent of the area to be processed; minx, miny, maxi, maxy; default is SADC region extent [11.0, -35.0, 58.5, 6.0]

The WMS was developed following the OGC specification version 1.3.0. It reads a configuration file (config.txt) to retrieve some default information required by the WMS, particularly the location of the default Mapfile.

Implemented WMS features include ***GetMap, GetFeatureInfo, GetCapabilities, GetLegendGraphic.***

The ***GetMap*** feature returns a graphic, in either PNG or GIF format, for specified map layers as listed in the LAYERS parameter.

The WMS uses the MapScript *mapObj.draw()* method to draw the specified map layers, which produces a MapScript *imageObj* object. The *imageObj* has a save() method which creates a graphic file in a specified image format.

Because the output images created by the 'imageObj.save()' method did not have transparency, it was necessary to reopen the output images and enforce the transparency as required by the WMS. The colour to be made transparent was picked from the Mapfile, where it was specified as the IMAGECOLOR. Image transparency was required to allow several map layers to be overlaid without obstructing each other.

The Java Advanced Imaging (JAI) API was used to load the image, creating a ***RenderedImage*** object. The colour model of the image is extracted using the method getColorModel() of the RenderedImage object. The colour model gives access to the colours available in the image. The colour which matches the *mabObj's imagecolor* is given an alpha value of 0 which makes it transparent. For each colour image colour model, there is an alpha component whose value ranges from 0 (or 0.0) to 255 (or 1.0). Zero (0) is for completely transparent while 255 is for completely opaque [44].

For the PNG image format, the Palette subclass of the PNGEncodeParam class [43] has methods to set the colours of a PNG image object and the transparency for the image. Once the transparency is set, the RenderedImage object is written back to file to overwrite the original (non-transparent) image. The transparency-enabled image is loaded again and drawn to an output stream associated with the HttpServletResponse object, which then displays it in the web browser.

GetFeatureInfo returns pixel information for the image (raster) data for a point specified by the client. The client chooses a format in which the information should be returned. The options are XML and HTML.

GetCapabilities returns an XML document showing the GIS layers available from the WMS and how to query or ask for services from the WMS. This was implemented because it is a standard WMS feature specified by OGC.

GetLegendGraphic was required to prepare a graphic legend for a GIS layer using the classifications in the Mapfile. The client specifies the GIS layer and also output image format as either PNG or GIF. The *GetLegendGraphic* feature is not a standard WMS feature and is not specified in the OGC WMS standard, but it is an important feature to implement since it helps the users interpret the GIS layers that they are viewing. The drawing of the map layer legend was implemented using the Java *BufferedImage* and *Graphics* classes. These two Java classes were required for image creation and drawing functions respectively.

The colours used in the drawing of the legend are obtained from the map layer object.

Because of the need to support time-series image handling, a ***TIME*** dimension was added to the WMS request. MapScript has support for temporal data through the use of the TILEINDEX in the Mapfile. The TILEINDEX is a file or database table that offers the three columns shown in Illustration 6.1.

Illustration 6.1

Data location / URL	***Data extent (polygon)***	***Timestamp***
....	....	....

The TILEINDEX is defined as a polygon (or *tilendex)* in the Mapfile. An example of a TILEINDEX layer that was used to obtain data from the PostGIS database is shown in Illustration 6.2.

Illustration 6.2

```
LAYER
  STATUS OFF
  NAME "time_idx"
  TYPE POLYGON
  PROJECTION
    "init=epsg:4326"
  END

  DATA "data_extent from (select image.image_id, datatype.data_extent, image.image_url,
image.time_period_start from datatype, imagetype, image where datatype.datatype_id=2 and
datatype.datatype_id=imagetype.image_datatype and
imagetype.imagetype_id=image.image_type) as sadcgeom using unique image_id using
SRID=5001"

  TEMPLATE "ttt_query.html"
  METADATA
    "wms_title" "TIME INDEX"
    "wms_srs"   "EPSG:4326"
    "wms_extent" "11.0 -35.0 58.5 6.0"
    "wms_timeextent" "2000-01-01/2008-04-21/P1D"
    "wms_timeitem" "time_period_start" #column in postgis table of type timestamp
```

```
  "wms_timedefault" "2000-01-01"
 END
 CONNECTION "dbname=rrsu_imagedb user=user1 password=pass1 port=5432
host=localhost"
 CONNECTIONTYPE postgis
END
```

As can be seen in illustration 6.2, the 'DATA' item points to an SQL statement, which will be processed by MapScript to return a table like the one shown in figure 6.5.

data_extent (geometry)	image_url (varchar)	time_period_start (timestamp)
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r07113.iml	2007-11-21 00:00:00
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r07112.iml	2007-11-11 00:00:00
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r07111.iml	2007-11-01 00:00:00
010300002089130000...	/home/blpt2/Documents/images/r00021.iml	2000-02-01 00:00:00
010300002089130000...	/home/blpt2/Documents/images/r00012.iml	2000-01-11 00:00:00
010300002089130000...	/home/blpt2/Documents/images/r00013.iml	2000-01-21 00:00:00
010300002089130000...	/home/blpt2/Documents/images/r00011.iml	2000-01-01 00:00:00
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r00021.iml	2000-02-01 00:00:00
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r00022.iml	2000-02-11 00:00:00
010300002089130000...	/media/hda2/RSData/rfe/GeoPrj/r00023.iml	2000-02-21 00:00:00

Figure 6.5: TILEINDEX table. 'data_extent' are polygons, the spatial extent of the images

When the WMS receives a TIME parameter, the value of the TIME parameter will be used to prepare a time filter to be applied on a table like the one in figure 6.5. An example time filter is shown below:

.. [WHERE] (date_trunc('day', time_period_start) = '2000-02-11')

This filter allows the right single record to be picked from the above TILEINDEX table or file. In order for the WMS to display the correct raster data, the Mapfile should also contain a layer of type 'RASTER' which points to the TILEINDEX shown in illustration 6.2. This RASTER layer must not contain a DATA item.

Invoking the WMS: Table 6.2 shows examples of how the client would invoke the web map server, and the result that would be returned by the web map server.

Table 6.2: Example of how to invoke the web map server

WMS query example	***Result***
http://blpt2-hpv:8000/servlet/mapserver.WMS?map=rainfall_timeseries.map&**LAYERS**=sadc_rainfall-wmst&**TRANSPARENT**=TRUE&TIME=0999-02-11&**FORMAT**=image%2Fpng&**STYLES**=default&**SERVICE**=WMS&**VERSION**=1.1.1&**REQUEST**=GetMap&**EXCEPTIONS**=application%2Fvnd.ogc.se_inimage&**SRS**=EPSG%3A4326&**BBOX**=-18.55,-55.5,88.05,26.5 &**WIDTH**=1040&**HEIGHT**=800	

6.2.5 WMS Client Implementation

The development of the WMS client was based on the MapBuilder open source application. Information used by the MapBuilder client is read from configuration and context documents. This information is static and does not readily support the need to display images from the database as specified by the user. The web page which loads MapBuilder carries the name of the configuration file, and the configuration file holds the name of the context document. Time-series image viewing was achieved using MapBuilder via a module called *MovieLoop*. The time at which to start and end the movie loop is read from the Mapfile referenced in the context document. This information is held in the following two lines in the *METADATA* section of the Mapfile.

```
"wms_timeextent" "2000-01-01/2008-04-21/P1D"
"wms_timedefault" "2000-01-01"
```

The *wms_timedefault* value tells the *MovieLoop* module where to start while *wms_timeextent* value controls the extents of the time span. For the system under development, it was necessary to allow the client to specify a specific time at which to start the time-series loop. This means that the user gets to see a specific image rather than start from the first image available in the time-series. The metadatabase schema has a *time_period_start* attribute giving the date at which the time-series loop should start. To pass the image date to the MovieLoop module, a variable *ImageTimestamp* was placed in the web page that holds the MapBuilder client, as shown in the following example.

```
<script>
...
var ImageTimestamp = '2002-02-01';
...
</script>
```

The following lines were added to the MovieLoop Javascript code:

```
if (window.ImageTimestamp){
   objRef.model.timestampList.childNodes[0].firstChild.nodeValue = ImageTimestamp;
if (this.timestampIndex!=null) {
   objRef.model.timestampList.childNodes[0].firstChild.nodeValue = ImageTimestamp;
 }  }
```

The above lines of code imply that if an *ImageTimestamp* variable was declared, it will be used to initialise the time (date) at which the time-series movie loop will start.

Because of the need to change the date of the images and the need to decide which configuration files to use with MapBuilder, a Java server page *(wms_viewer)* was developed to handle the dynamic part and load the MapBuilder WMS client. This JSP takes the image ID of a specified image and use it to extract information about the image from the database. Among the extracted information is the time_period_start, which is used to initialise the *ImageTimestamp* variable.

The *wms_viewer* JSP also picks the right configuration file to use based on the image data type to which the image belongs, as shown below.

```
............
        if(imagetype_name.equalsIgnoreCase("CPC_RFE_SADC"))
                out.print("var mbConfigUrl='ts_SADC_RFE_config.xml';");
        if(imagetype_name.equalsIgnoreCase("NOAA_NDVI_SADC"))
                out.print("var mbConfigUrl='ts_SADC_NOAANDVI_config.xml';");
..............
```

Template Mapfiles were prepared for the main image data types included in the database. To allow for the automatic preparation of the legends from the image data using the *GetLegendGraphic* feature of the web map server, classifications of the image data were added to the Mapfiles. The image classification tells the web map server which colours to use for all the values in the image. The classifications have to cover all the values in the image. Without these classifications, the images will only be displayed as grey level pictures. These classifications were done for all the main image types held in the database, based on information from working papers used by the RRSU.

Configuration files and context files were prepared for the main image types shown in table 6.3 below.

Table 6.3 Image types that were tested in the WMS client implementation

Image Data type	*Grid cell size*	*Extent (minx, miny, maxx, maxy)*
SADC-wide Rainfall 10-day images	0.1 degrees	11.0, -35.0, 58.5, 6.0
Africa-wide Rainfall 10-day images	0.1 degrees	-24.6, -42.243, 64.523, 43.712
SADC-wide Vegetation 10-day maps	0.1 degrees	11.0, -35.0, 58.5, 6.0
SADC-wide SPOT Vegetation 10-day maps	0.00892 degrees	11.0, -35.0, 58.5, 6.0
Africa-wide Vegetation 10-day maps	0.1 degrees	-24.6, -42.243, 64.523, 43.712

6.3 Image Analysis Functionality Implementation

6.3.1 Reference Images

Some image analysis requires the presence of references images, which are usually long or short term averages (sometimes called normals) of images in a time-series collection. Reference images are particularly useful in the analysis of images in the food security early warning field. For example, images are often compared with reference images to determine whether the parameter being depicted by the image (e.g. rainfall) is below or above that expected for the time period in question. For images representing averages of the time series, the year cannot be specified when recording their dates. For example, image data for the period January 21-31 of 2008 will be compared with the average data for the period 21-31 January over all the years for which that particular image data exists.

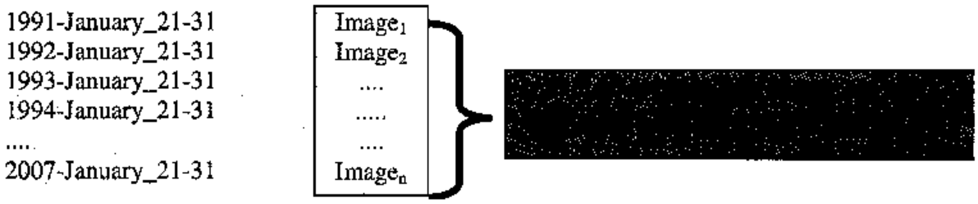


Figure 6.6: Average Images

In order to facilitate the insertion of these images into the image metadatabase, the year '0999' was chosen for all averages. For example in the case of January 21-31 the image is given the following timestamps:

period start: 0999-01-21; period end: 0999-01-31

To allow the entry of average images into the metadatabase, the image entry form allows user to specify whether the image is an average or not. The image entry JSP code was also modified such that the average images are given the year '999', when the user specifies that the image is an average image.

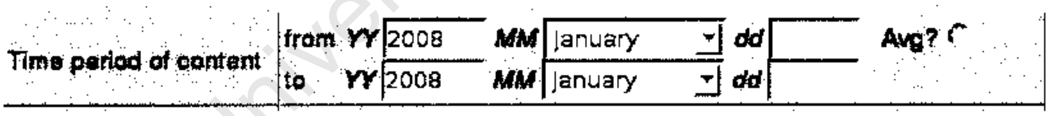


Figure 6.7: Section of image metadata entry form with radio button 'Avg?'

In the interface above, the users only need to specify the month, chosen from the drop-down lists, and the day(s) typed in the *"dd"* boxes, and click the *"Avg?"* radio button to indicate that the image is an average or reference image.

6.3.2 Java WinDisp Class

The majority of the images to be held in the metadatabase are in the WinDisp (or IDA

format that could be used to retrieve information from these images for use in image analysis and related purposes. So it was necessary to prepare a Java class that can read this image format.

Two main constructors were prepared for the Java WinDisp class, shown below:

1. WinDisp(String sImage_Title, ***byte*** *image_type,* ***byte*** *image_projection,* ***short*** *image_height,* ***short*** *image_width,* ***double*** *latitude_center,* ***double*** *longitude_center,* ***double*** *xcenter,* ***double*** *ycenter,* ***double*** *pixelsize_x,* ***double*** *pixelsize_y,* ***double*** *parallel1,* ***double*** *parallel2,* ***byte[]*** *pixels)*

2. WinDisp(String sWinDispFilename)

The first constructor creates a new WinDisp image object using the supplied parameters and the pixel data provided. The second constructor creates a WinDisp image object from the file whose name is supplied.

Selected methods of the WinDisp class that were developed to retrieve information from the images and to write image data to file are shown in table 6.4.

Table 6.4 Selected methods of the WinDisp class

Method name	*Purpose of method*
short getWidth()	Provides the width of the image
short getHeight()	Provides the height of the image
byte[] getPixels()	Provides the pixel data of the image
int getProjection()	Provides the projection used in the image
double getDx()	Pixels size in the East-West (X) direction
double getDy()	Pixels size in the North-South (Y) direction
double getXcenter()	Provides pixel number that coincides with the Longitude centre of the image
double getYcenter()	Provides pixel number that coincides with the Latitude centre of the image
double getLongCenter()	Provides central longitude of the images projection
double getLatCenter()	Provides central latitude of the images projection
double getParallell()	Provides 1st standard parallel
double getParallel2()	Provides 2nd standard parallel
double getMaxLat()	Provides the maximum latitude of the image, or the northern bound
double getMinLong()	Provides the minimum longitude of the image, or the western bound
WriteIDA(String sIdaFile)	Writes the WinDisp image object to file with the specified file name

6.3.3 Java TimeSeries class

The TimeSeries Java class was developed to hold the information for a time-series and prepare required analyses on the images in the time-series. The TimeSeries object holds a list of image URLs and had methods to create the following:

4. **Average** – an image representing the average of the images in the time-series
5. **Sum** – an image representing the sum of the images in the time-series
6. **Percentage** – an image representing the sum of the images in the time-series as a comparison of the sum of the corresponding reference images.

The TimeSeries Java class has the following constructors:

- **TimeSeries() :** creates a new blank *TimeSeries* object
- **TimeSeries(String sFileList) :** creates a *TimeSeries* object using the list of image URLs in the supplied file
- **TimeSeries(ArrayList<String> inList) :** creates a *TimeSeries* object using a list of image URLs.

The methods used by the *TimeSeries* class are shown in table 6.5.

Table 6.5: Methods of the TimeSeries class

Method	***Purpose***
addItem(String newItem)	*Insert a new Image URL into the TimeSeries list*
removeItem(int index)	*Remove the image URL at the position specified by the index from the TimeSeries list*
int getSize()	*Return the number of Image URLs in the TimeSeries list*
String getItem(int index)	*Return the image URL at position 'index' in the image list*
WinDisp getAverage()	*Get an image object which represents the average of the images in a specified list*
WinDisp getSum()	*Get an image object which represents the sum of the images in a specified list*
WinDisp getPercent(ArrayList<String> refList)	*Prepares a WinDisp image object that represents a comparison of the specified images with the reference images*

The image analysis functions chosen were those most used in a remote sensing based early warning monitoring system for food security. Some of these are briefly explained in section 2.8.

The development of the image analysis functions was based on the image arithmetic operators which are offered by the **Java Advanced Imaging (JAI)** API [43]. The following operators were used:

Add – add two or more images

Divide – divide one image by another

DivideByConst – divide image by a constant

MultiplyConst – multiply image by a constant.

The image analysis methods of the TimeSeries Java class are described below:

getAverage() method Prepares average of the images in the *TimeSeries* object URL list as shown in Figure 6.8. The *getAverage()* method uses URLs in the TimeSeries list to create WinDisp objects with each of the URLs.

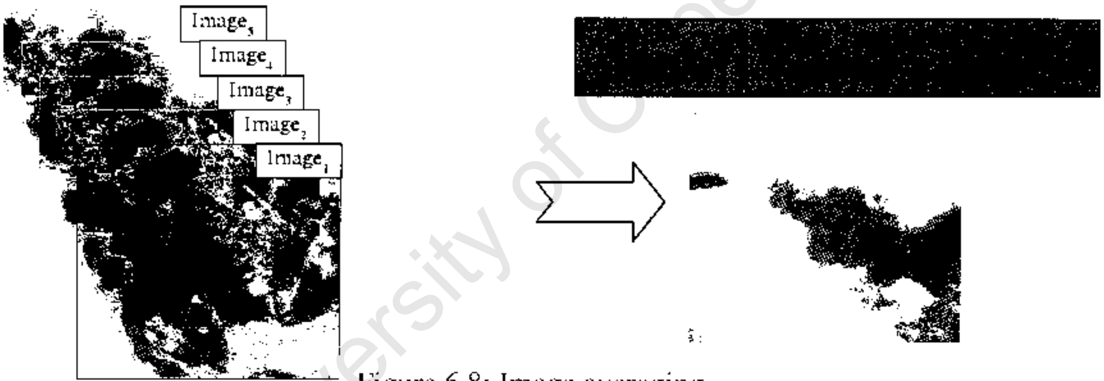


Figure 6.8: Image averaging

Using the *getWidth()*, *getHeight()* and *getPixels()* methods, Java-recognised image types can be created, which include the following:

— RenderedImage [49]

— TiledImage [50]

— BufferedImage [48].

These Java image types have a setData() method used to transfer the image data from the WinDisp image object to the Java image.

The JAI compatible image types are then used in the image arithmetic JAI operations to get the average image. The ***Add*** and ***DivideByConst*** JAI operators are used. When the average image is obtained via JAI, this image's pixels are used to create a new resultant WinDisp image object, which is then written to file using the *WriteIDA(String sldaFile)* method of the WinDisp class.

getSum() method Image series sum = (Image$_1$ + Image$_2$ + ... + Image$_n$). Follows same procedure as described above for the average but uses only the ***Add*** JAI operator.

getPercent() method This operation was carried out on a pixel by pixel basis using Java arrays. Pixel data was retrieved from the WinDisp images using the *getPixels()* method and used in the formula shown in figure 6.9.

Total of Image series as percentage of the total of the averages ⇒

$$\frac{100 \times (Image_1 + Image_2 + \ldots + Image_n)}{(AvgImage_1 + AvgImage_2 + \ldots + AvgImage_n)}$$

Figure 6.9: Formula used to calculate percentage of average images

This operation produces a single image providing a comparison between the total of the time-series images in the specified period and the total of the average (or reference) images for the period, illustrated in figure 6.9. In the figure, *AvgImagel* is the reference image for the period represented by *Image].* In the resultant image, the pixel values are percentages which can be presented with appropriate colour coding in the web map service.

6.3.4 ImageSeries Java Servlet

A Java Servlet, ImageSeries, was prepared to capture the information required by the TimeSeries class and invoke the TimeSeries class' functions. The roles of the ImageSeries servlet are listed below:

- to create a TimeSeries object and invoke relevant methods of this object
- to build a list of time-series dates *(Time-series list builder)*
- to connect to the metadatabase to retrieve image ID and image URLs given an image type and timestamp
- to retrieve the reference or average images in case comparisons are required.

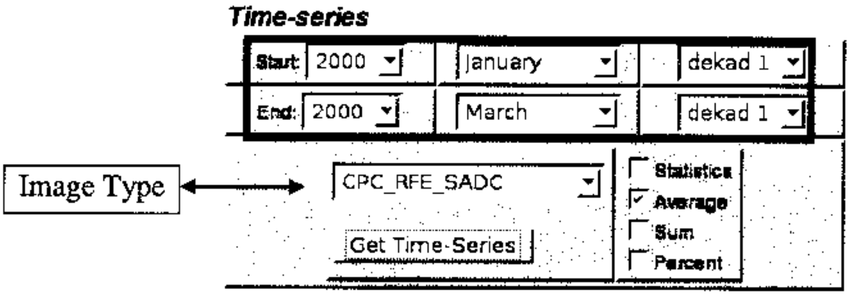


Figure 6.10: Interface for time-series image analysis

In figure 6.10, *dekad* 1 represents the first ten days of the month, *dekad 2* period 11-20 of the month, while *dekad 3* is the period 21 to the end of the month. Pressing the 'Get Time-Series' button invokes the *ImageSeries* Servlet.

Time-series list builder is a function of the ImageSeries Servlet that builds a list of timestamps at dekadal (10 day) interval. It receives a pair of start and end dates and returns a list of timestamps like the ones shown in the 'Timestamps' column in table 6.6 below.

Table 6.6 Information retrieved for time-series image collections

Timestamps	*Image_ID*	*Image URL*	*URL for Reference / Average Image*
2000-01-01	11	/home/blpt2/Documents/images/r00011.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav011.iml
2000-01-11	12	/home/blpt2/Documents/images/r00012.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav012.iml
2000-01-21	13	/home/blpt2/Documents/images/r00013.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav013.iml
2000-02-01	14	/media/hda2/RSData/rfe/GeoPrj/r00021.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav021.iml
2000-02-11	15	/media/hda2/RSData/rfe/GeoPrj/r00022.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav022.iml
2000-02-21	16	/media/hda2/RSData/rfe/GeoPrj/r00023.iml	/media/hda2/RSData/rfe/Average/ARC_10yr/rfav023.iml

These timestamps are used with the image type to retrieve the IDs of the associated images and their URLs. The reference images' URLs are also retrieved in case they will be required in the ensuing analysis.

The list of images (as in the Image URL column in table 6.6 above) is used to create a *TimeSeries* object. The *TimeSeries* object then holds a list of the images which will be used by the *getAverage(), getSum() or getPercent()* methods as required by the type of analysis. These methods produce a WinDisp image object. This object is then written to file and used in a GIS layer to be displayed by the web map server.

6.3.5 Integration of image analysis results with the WMS

As mentioned earlier, the image analysis produces an image file in the WinDisp format. For this image to be displayed in the web map service, the ImageSeries Servlet was extended to create the following three temporary files:

— Mapfile that carries the file name of the image created via the image analysis

— A web context file which carried the name of the Mapfile above

— A configuration file which tells MapBuilder to use the web map context file above

The creation of the Mapfile and the web map context uses information retrieved from the metadatabase, such as the geographic extent of the images, the coordinate reference system and the grid cell size of the images.

The ImageSeries Servlet then redirects the user to the WMS viewer where it forwards the name of the configuration file in use. This was done using the following code.

```
String sRedirectURL = sServerURL+"wms/wms_viewer2.jsp?config="+ configfileURL;
response.setStatus(response.SC_MOVED_TEMPORARILY);
response.setHeader("Location", sRedirectURL);
```

Chapter 7 — Evaluation

This chapter discusses the steps taken to evaluate the prototype system and the results of the evaluation.

7.1 Prototype System Testing and Evaluation by Users

The prototype system developed was presented to the users in a live demonstration. The presentation focused on aspects of image data management and analysis which the users were familiar with and was generally well understood, judging by the comments that followed. Users were also given the chance to access the system from their own computers via the Intranet. The users provided comments on the database management interface, the web map interface used for viewing the images and the functionality provided for image analysis.

7.2 Task Performance using the Prototype System

Three users were observed as they carried out the main tasks which are covered by the prototype developed. The users were expert users and data-managers from RRSU. The tasks carried out are listed below:

- Task 1 - Searching and displaying images
- Task 2 - Preparing image averages over a specified period of time
- Task 3 - Preparing images that represent percentages of reference images

These activities were carried out using the prototype and also using WinDisp, the main software package that is currently used to do these tasks. Raster datasets are widely used in the early warning community and the WinDisp software is the main software used for the handling of these datasets. The users already had a good working knowledge of the WinDisp software. The tasks were broken down into sub-tasks and the amount of time spent by each of the users to complete the sub-tasks was observed and summarized.

Each task is carried out while the prototype system and WinDisp are already loaded, so the times measured do not include the time for starting the prototype or for loading WinDisp. For some of the tasks, the WinDisp system had to be reloaded, so that the software would not have memory of the images already used or the file folders previously browsed.

The tasks were performed on the same computer by all the three users. The computer had the Windows Vista operating system, and had the following specifications: Intel(R) Core(TM)2 Duo CPU P8400 @ 2.26GHz 2.27 GHz, memory (RAM): 2.00GB.

7.2.1 Task 1 - Searching and displaying an image

For the two options under comparison, WinDisp and the prototype system, this task was broken down into subtasks that were performed to accomplish the task and the time taken to complete each of these subtasks is measured in seconds.

For the display of images using WinDisp, the users were given documents showing them where to find the images and colour tables that they had to use. The users were given some time to familiarise with the folder structures before they were timed as they displayed the images. Each user had to display 5 different images and the software was restarted on each attempt, so that the users would always start from scratch.

The amounts of time required to complete sub-tasks were recorded in seconds and the results for each of the 3 users who participated in the evaluation are shown in table 7.1.

Table 7.1: Measured time for displaying images using WinDisp

Sub-task	User	Time (seconds)							
		t1	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
1. browse for the image	1	23.0	22.0	25.0	23.0	23.0	23.2	23.0	1.1
	2	18.0	15.0	18.0	22.0	17.0	18	18.0	2.5
	3	13.0	17.0	18.0	14.0	22.0	16.8	17.0	3.6
2. browse for the colour table	1	10.0	10.0	9.0	9.0	8.0	9.2	9.0	0.8
	2	7.0	7.0	8.0	7.0	7.0	7.2	7.0	0.4
	3	7.0	8.0	7.0	9.0	8.0	7.8	8.0	0.8
3. load image	1	1.0	1.0	1.0	1.0	2.0	1.2	1.0	0.4
	2	1.0	1.0	1.0	1.0	1.0	1	1.0	0.0
	3	1.0	1.0	1.0	1.0	1.0	1	1.0	0.0
Total Time	1	34.0	33.0	35.0	33.0	33.0	33.6	33.0	0.9
	2	26.0	23.0	27.0	30.0	25.0	26.2	26.0	2.6
	3	21.0	26.0	26.0	24.0	31.0	25.6	26.0	3.6

The average time for the three users is shown in table 7.2 below.

Table 7.2: Average time for searching and displaying images using WinDisp

Task	*Average time (s)*	*Comment*
1. browse for the image	19.3	Time varies depending on folder structure
2. browse for the colour table	8.1	Time varies depending on folder structure
3. load image	1.1	Depends on the size of the image file
Total Average Time	*28.5*	
Overall Median	*27*	
Overall Std Dev	*4.5*	

For the display of the images using the prototype, the users were given time to familiarise themselves with the interface before their efforts in displaying the images were timed. Three users were timed and the measured time is shown in table 7.3 below.

Table 7.3: Measured time for displaying image using the developed prototype

Sub-task	User	Time (seconds)							
		t1	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
1. specify the period	1	7.0	5.0	5.0	5.0	5.0	5.4	5.0	0.9
	2	5.0	7.0	5.0	6.0	5.0	5.6	5.0	0.9
	3	5.0	6.0	6.0	5.0	5.0	5.4	5.0	0.5
2. specify the type of image	1	1.0	1.0	2.0	2.0	1.0	1.4	1.0	0.5
	2	2.0	2.0	3.0	2.0	2.0	2.2	2.0	0.4
	3	2.0	2.0	2.0	3.0	2.0	2.2	2.0	0.4
3. load image	1	6.0	5.0	4.0	4.0	4.0	4.6	4.0	0.9
	2	4.0	5.0	4.0	5.0	3.0	4.2	4.0	0.8
	3	5.0	4.0	5.0	4.0	4.0	4.4	4.0	0.5
Total Time	1	14.0	11.0	11.0	11.0	10.0	11.4	11.0	1.5
	2	11.0	14.0	12.0	13.0	10.0	12	12.0	1.6
	3	12.0	12.0	13.0	12.0	11.0	12	12.0	0.7

Table 7.4: Average time for displaying images using the developed prototype

Task	***Average time (s)***	***Comment***
1. specify the period	5.5	
2. specify the type of image	1.9	
3. load image	4.4	The WMS client takes a bit of time to load the resources required to display the image
Total Average Time	*11.8*	
Overall Median	*12*	
Overall Std Deviation	*1.3*	

The task of searching for images and displaying them takes on average 41 % of the time taken when using the WinDisp software. The time savings are made especially on the part of searching for the images.

7.2.2 Task 2 - Preparing Image averages

WinDisp is one of the tools commonly used for this purpose which is publicly available to the early warning practitioners. The task of preparing averages with WinDisp does require some skill on the part of the users. Even though some of the users were already aware of the steps taken to create these averages, the steps were written down for them and there were given time to familiarise with the steps. The users were given documents which showed which averages were to be created and where to find the input images. The images used were among the ones used in the earlier exercise where images were displayed.

Table 7.5: Measured time for preparing image averages using WinDisp

Sub-task	User	Time (seconds)							
		t1	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
1. Browse to folder containing the images	1	18.0	15.0	13.0	14.0	14.0	14.8	14.0	1.9
	2	17.0	17.0	16.0	15.0	18.0	16.6	17.0	1.1
	3	15.0	17.0	19.0	14.0	16.0	16.2	16.0	1.9
2. Select the images needed	1	6.0	5.0	5.0	6.0	5.0	5.4	5.0	0.5
	2	7.0	6.0	6.0	6.0	5.0	6	6.0	0.7
	3	5.0	5.0	6.0	5.0	6.0	5.4	5.0	0.5
3. Save the list of images as text file (specify file name and folder)	1	12.0	16.0	14.0	14.0	12.0	13.6	14.0	1.7
	2	17.0	11.0	14.0	12.0	15.0	13.8	14.0	2.4
	3	11.0	11.0	13.0	12.0	16.0	12.6	12.0	2.1
4. Specify the output image name and path	1	12.0	15.0	11.0	16.0	15.0	13.8	15.0	2.2
	2	16.0	15.0	17.0	14.0	16.0	15.6	16.0	1.1
	3	13.0	16.0	14.0	14.0	14.0	14.2	14.0	1.1
5. Display the image	1	27.0	22.0	22.0	20.0	21.0	22.4	22.0	2.7
	2	28.0	28.0	24.0	25.0	25.0	26	25.0	1.9
	3	26.0	25.0	26.0	26.0	22.0	25	26.0	1.7
Total Time	1	75.0	73.0	65.0	70.0	67.0	70	70.0	4.1
	2	85.0	77.0	77.0	72.0	79.0	78	77.0	4.7
	3	70.0	72.0	78.0	71.0	74.0	73	72.0	3.2

Table 7.6: Average time for preparing Image averages using WinDisp

Task	*Average time (s)*	*Comment*
Browse to folder containing the images	15.9	Amount of time would depend on folder structure
Select the images needed	5.6	Building a list of images
Save the list of images as text file (specify file name and folder)	13.3	Amount of time would depend on folder structure
Specify the output image name and path	14.5	Time would depend on folder structure
Display the image	24.5	Most time is spent browsing to the folder where image was saved
Total Average Time	*73.7*	
Overall Median	*73*	
Average Std Deviation	*5.1*	

The table below show the measured time when using the prototype system to prepare averages of time-series images. These tasks were demonstrated to the users and the users were given time to practise creation of averages before they were timed.

Table 7.7: Measured time for preparing image averages using the developed prototype

Sub-task	User	Time (seconds)							
		t1	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
1. specify the period	1	7.0	7.0	9.0	9.0	8.0	6.8	8.0	1.0
	2	6.0	9.0	6.0	6.0	8.0	6.2	6.0	1.4
	3	7.0	5.0	7.0	7.0	9.0	6.3	7.0	1.4
2. specify the type of image	1	1.0	1.0	2.0	1.0	2.0	1.3	1.0	0.5
	2	2.0	1.0	1.0	1.0	1.0	1.3	1.0	0.4
	3	1.0	1.0	1.0	2.0	1.0	1.5	1.0	0.4
3. Load image viewer	1	4.0	3.0	3.0	3.0	3.0	2.8	3.0	0.4
	2	3.0	5.0	3.0	3.0	4.0	3.3	3.0	0.9
	3	3.0	3.0	3.0	4.0	3.0	3.2	3.0	0.4
Total Time	1	12.0	11.0	14.0	13.0	13.0	12.6	13.0	1.1
	2	11.0	15.0	10.0	10.0	13.0	11.8	11.0	2.2
	3	11.0	9.0	11.0	13.0	13.0	11.4	11.0	1.7

Table 7.8: Average time for Preparing Image averages using the developed prototype

Task	*Average time (s)*	*Comment*
1. specify the period	7.3	
2. specify the type of image	1.3	
3. Load image viewer	3.3	Includes time for calculating average and loading image in browser
Total Average Time	*11.9*	
Overall Median	*12*	
Overall Std Deviation	*1.7*	

For preparing image averages, users agreed that the prototype system simplifies the task for them significantly and also saves them time. The experiments showed that users could save up to 84% of their time when using the prototype, compared to WinDisp.

7.2.3 Task 3 - Preparing percentage of average or reference images

Table 7.9: Measure time for preparing percentage of average images using WinDisp

Sub-task	User								Time (seconds)
		t1	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
1. Browse to folder	1	17.0	16.0	19.0	17.0	15.0	16.8	17.0	1.5
containing the images	2	19.0	17.0	20.0	15.0	17.0	17.6	17.0	1.9
	3	16.0	18.0	17.0	15.0	19.0	17	17.0	1.6
2. Select the images needed	1	6.0	6.0	5.0	6.0	4.0	5.4	6.0	0.9
	2	5.0	6.0	6.0	6.0	5.0	5.6	6.0	0.5
	3	5.0	5.0	4.0	6.0	5.0	5	5.0	0.7
3. Save the list of images	1	14.0	17.0	17.0	20.0	16.0	16.8	17.0	2.2
as text file (specify file	2	18.0	12.0	16.0	14.0	15.0	15	15.0	2.2
name and folder)	3	16.0	17.0	15.0	12.0	16.0	15.2	16.0	1.9
4. Specify the output image	1	17.0	16.0	17.0	15.0	15.0	16	16.0	1.0
name and path	2	16.0	16.0	13.0	18.0	14.0	15.4	16.0	1.9
	3	14.0	12.0	17.0	13.0	15.0	14.2	14.0	1.9
Preparing Reference Images									
5. browse to folder	1	20.0	19.0	23.0	20.0	22.0	20.8	20.0	1.6
containing the reference	2	22.0	24.0	21.0	22.0	22.0	22.2	22.0	1.1
images	3	17.0	21.0	20.0	19.0	19.0	19.2	19.0	1.5
6. select the reference	1	7.0	7.0	8.0	7.0	6.0	7	7.0	0.7
images needed	2	11.0	9.0	10.0	10.0	8.0	9.6	10.0	1.1
	3	6.0	5.0	7.0	8.0	6.0	6.4	6.0	1.1
7. save the list of	1	15.0	13.0	15.0	15.0	14.0	14.4	15.0	0.9
references images as text	2	16.0	15.0	18.0	16.0	14.0	15.8	16.0	1.5
file	3	14.0	14.0	15.0	15.0	13.0	14.2	14.0	0.8
8. Specify the output	1	16.0	16.0	16.0	15.0	16.0	15.8	16.0	0.4
reference image name and	2	15.0	17.0	17.0	16.0	17.0	16.4	17.0	0.9
Path	3	16.0	15.0	15.0	15.0	16.0	15.4	15.0	0.5
Performing Image algebra									
9. specify formula	1	10.0	11.0	11.0	10.0	12.0	10.8	11.0	0.8
(100*A/B)	2	10.0	12.0	10.0	11.0	11.0	10.8	11.0	0.8
	3	9.0	11.0	11.0	10.0	10.0	10.2	10.0	0.8
10. specify output file	1	14.0	11.0	13.0	13.0	12.0	12.6	13.0	1.1
name	2	12.0	12.0	11.0	14.0	12.0	12.2	12.0	1.1
	3	11.0	12.0	14.0	12.0	12.0	12.2	12.0	1.1
11. browse for the quotent	1	12.0	16.0	15.0	15.0	13.0	14.2	15.0	1.6
image file	2	14.0	12.0	12.0	13.0	12.0	12.6	12.0	0.9
	3	13.0	13.0	14.0	12.0	11.0	12.6	13.0	1.1
12. browse for the divident	1	11.0	11.0	12.0	10.0	10.0	10.8	11.0	0.8
image file	2	12.0	12.0	12.0	11.0	12.0	11.8	12.0	0.4
	3	11.0	10.0	12.0	12.0	11.0	11.2	11.0	0.8
13. Display the resultant	1	22.0	19.0	19.0	20.0	21.0	20.2	20.0	1.3
image	2	21.0	23.0	20.0	23.0	22.0	21.8	22.0	1.3
	3	18.0	19.0	19.0	21.0	18.0	19	19.0	1.2
Total Time	1	181.0	178.0	190.0	183.0	176.0	181.6	181.0	5.4
	2	191.0	187.0	186.0	189.0	181.0	186.8	187.0	3.8
	3	166.0	172.0	180.0	170.0	171.0	171.8	171.0	5.1

Table 7.10: Average time for preparing percentage of average images using WinDisp

Task	*Average time (s)*	*Comment*
Processing the images summation		
browse to folder containing the images	17.1	Amount of time depends on file folder structure
select the images needed	5.3	Building time series list
save the list of images as text file (specify filename and folder)	15.7	Amount of time depends on file folder structure
Specify the output image name and path	15.2	Amount of time depends on file folder structure
Processing the Reference images summation		
browse to folder containing the reference images	20.7	Amount of time depends on file folder structure
select the reference images needed	7.7	Amount of time depends on file folder structure
save the list of references images as text file	14.8	Amount of time depends on file folder structure
Specify the output reference image name and path	15.9	Amount of time depends on file folder structure
Performing Image algebra		
specify formula (100*A/B)	10.6	
specify output file name	12.3	
browse for the quotent image file	13.1	Amount of time depends on file folder structure
browse for the divident image file	11.3	Amount of time depends on file folder structure
Display the resultant image	20.3	
Total Average Time	*180.1*	
Overall Median	*181.0*	
Average Std Deviation	*7.8*	

Table 7.11: Measure time for preparing percentage of average images using the prototype

Sub-task	User	*tl*	*t2*	*t3*	*t4*	*t5*	*Ave*	*Median*	*Std Dev*
								Time (seconds)	
1. specify the period	1	6.0	6.0	6.0	5.0	5.0	5.6	6.0	0.5
	2	5.0	5.0	6.0	6.0	6.0	5.6	6.0	0.5
	3	4.0	5.0	6.0	6.0	5.0	5.2	5.0	0.8
2. specify the type of image	1	1.0	2.0	1.0	1.0	1.0	1.2	1.0	0.4
	2	2.0	1.0	1.0	1.0	1.0	1.2	1.0	0.4
	3	2.0	1.0	2.0	2.0	1.0	1.6	2.0	0.5
3. display resultant image	1	4.0	4.0	4.0	3.0	4.0	3.8	4.0	0.4
	2	3.0	4.0	6.0	4.0	4.0	4.2	4.0	1.1

	3	3.0	2.0	3.0	3.0	3.0	2.8	3.0	0.4
Total Time	1	11.0	12.0	11.0	9.0	10.0	10.6	11.0	1.1
	2	10.0	10.0	13.0	11.0	11.0	11	11.0	1.2
	3	9.0	8.0	11.0	11.0	9.0	9.6	9.0	1.3

Table 7.12: Average time for preparing percentage of average images using the prototype

Task	*Average time (s)*	*Comment*
1. specify the period	5.5	
2. specify the type of image	1.3	
3. display resultant image	3.6	The WMS client takes a bit of time to load the resources required to display the image
Total Average Time	*10.4*	
Overall Median	*11.0*	
Overall Std Deviation	*1.3*	

It can be seen from tables 7.10 and 7.12 that this task involves more steps using WinDisp than it does using the developed prototype system, which automates most of the steps. Even though WinDisp allows the automation of certain tasks through the use of text command files, this procedure is not possible to automate because of the list building part which requires manual selection of the files for use in the task. Using the developed prototype, users took on average only 6% of the time taken when they used WinDisp.

7.2.4 Discussion

While the users managed to perform the tasks without much difficulty, some appeared to struggle to get back to their starting point for each task since the prototype opens a new browser window to display the results of each of the tasks. For example, each image is displayed in a new window and the results of image analyses are also displayed in new windows. This meant that users always had to look for the window with the homepage of the prototype when they had to do a new task.

Another observation was that 2 out of the 3 users preferred having the month first, the dekad, and then the year, on the interface for specifying the dates or periods for the images.

Overall, observations of the users showed that the users could make use of the prototype facility with minimal assistance.

The results of the experiments outlined above shows that the prototype system saves time for all the users who participated in the system, even though some of them had a lot of experience in using WinDisp, which is the system that has been used traditionally.

7.3 Comments on the Prototype System

The main comments given by the users are outlined in the following sections.

7.3.1 Comments on Database Management Interface

General Impressions

The interface was generally well received by the users, who felt that the interface allowed the entry of adequate information about the images held by the RRSU. Issues raised by the users were mainly about the clarity of the terminology used on the interface, the need for extra information or features and the ease of use of the database management interface.

Improvements and extensions

Users suggested that raster data parameters which are required when creating an image type metadata record should be read from sample raster data files automatically instead of being entered manually. They argued that this would reduce the chances of providing wrong information and also reduce workload on the part of the data managers.

An automated estimation routine was also suggested for images which have contamination, for example cloud cover. The percentage of cloud cover could be estimated by opening the files and counting affected pixels.

In terms of extra features required, some users expressed the need for the following:

- A facility that allows them to draw a bounding box on a map to specify areas of interest.
- User access control so that different types of users will have different privileges when accessing the system. For example, general users or the public will have access to the data discovery and image analysis facilities but not the data management interface, which would be restricted to a smaller set of users, or database managers.
- More image formats to be included or accommodated in the system. A download facility when the image search results are presented to allow the user to copy the data onto their own computers.
- A tool to create image overviews or thumbnails automatically so that these can be entered into the metadatabase without much effort on the part of the data managers.

Likes and Dislikes

Users appreciated the drop-down lists offered on some of the metadata entry forms which help users choose appropriate entries. Users felt that even though they had good prior exposure to the terminology used on the entry forms, some of the terms needed to be changed as they could be interpreted differently by different users, for example, distinguishing between the terms "image title" and "image name" was not straightforward for some users.

7.3.2 Comments on Web Map Service

General Impressions

The general impression was that the web-based image viewer was a good start and carries a lot of promise. Users agreed that the web map service would even allow less technical viewers to analyse image data with little difficulty.

Improvements and extensions

Users expressed the need for flexibility in the use of colour schemes for the display of images. The prototype only offered the use of one colour scheme for each image type, but users indicated they may want to use colours that are different from the ones commonly used, for example, if they wish to highlight certain ranges of values in the image by using a more prominent colour.

Likes and Dislikes

When navigating through a time-series of images using the "next image" or "previous image" tools on the interface, the WMS always presents the full extent of the next/previous image even though the user may only be interested in a specific sub-region of the image. The users preferred that information on the currently selected region be passed to the WMS such that it shows only the same region when the new image is displayed.

7.3.3 Comments on future use of the prototype system

The targeted users gave their reactions on whether they would use the prototype system in their work routines, and how they would use it. Two user reactions were given on this question, as follows.

User reaction 1: "The WMS image viewer is a good addition to the image review meetings that are held by the RRSU during the main rainfall season, where WinDisp has been used traditionally for the display and interrogation of images. The time savings realised through the use of the prototype system would help shorten some of these meetings, so that staff could have more time for other activities."

User reaction 2: "The prototype system could be used to respond to requests for images from SADC nationals, who either want graphic overview of the images or they want the image data in a suitable GIS format. In its current state the prototype would adequately account for the users who just want to view the image data or copy it from the screen into their documents, but the download facility needs to be improved to meet custom requests from users in the SADC region."

7.3.4 Comments from users outside RRSU

In order to get impressions on how usable and relevant the prototype was to users outside the targeted department, two external users were chosen to give their opinions on the RRSU metadatabase system prototype. These users were chosen for their experience with user interfaces and information systems. One was an information officer in his organization and the other was a geographical information systems analyst. These users were given a general idea of what the prototype system sought to address and a short introduction of the prototype web pages.

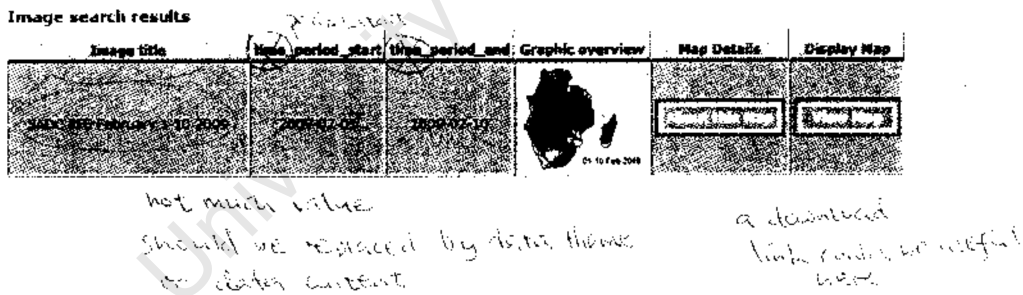


Figure 7.1: Sketch showing comments given on the search results presentation

Comment: "On the search results presentation, the 'image titles' do not seem to count for much, perhaps there should be 'theme' or human-readable 'data content', followed by the period information and the rest as shown in the figure 7.1. At the end there could be a link which allows users to download the image data files."
Comment: "On the 'Image Search Form' (Figure 7.2), things should be moved around such that the more understandable terms like "period", "theme", "keywords" should be presented first to the user and it should be made clear to the users that a search can be successfully

completed on the basis of these, without the other information being provided. Users should understand that there are items which are optional so that they do not bother trying to understand them or to supply information for them."

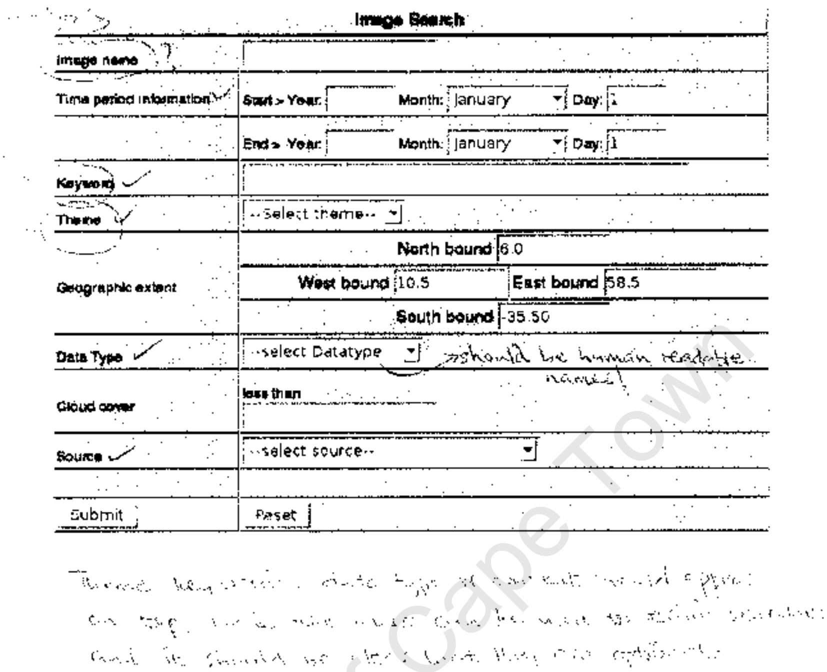


Figure 7.2: Sketch showing some of the comments given on the image search form

Comment: "The term 'browse graphic' is not a common term and should be replaced by `thumbnail' or something more common."

Comment: "The icon, if , used to represent the link to the data dictionary is a bit distracting on the metadata entry forms and should be de-emphasised as much as possible. Pressing the help icon should pop up a small window with the information required instead of a new web-page with the full data dictionary."

Comment: "Some of the more common database management functions for example, the `image entry' facility, could be accessed directly from the homepage of the prototype. The other database management functions could be on another page which can be accessed through a link on the homepage."

Comment: "I would not mind having to click twice to get to the other features offered, like descriptions of the data types held in the database. In view of this, it would be important to ensure that space on the homepage is reserved for only the most important features, and other features should be provided to make the homepage more attractive. For example users may want to see one of the more interesting images or maps extracted from the meta-database, on the front page. On maybe show the latest images available. The homepage should be restructured to allow for these inclusions."

Comment: "There should a web page that describes the RRSU metadatabase, and lists some frequently asked questions. For example, information on how other users can contribute to the database, if they can at all. This web page would be the one that carries information about the data types and image types held in the RRSU image collection."

Comment: "A glossary would also be useful in explaining some less apparent features or terms used. All in all, there should be more information to guide the user on how to use the meta-database system, and contribute to the system, instead of just viewing and analysing the images."

7.3.5 Discussion

The comments showed that users within RRSU focused more on the ease of use of the prototype and the functionality that it provides. The targeted users gave useful comments on improvements that could be made to the prototype to make it a solid part of their work routines. Comments from outside users focused on general aesthetics and the clarity of the terms used in the metadatabase and on the interface. Their comments were meant to make the interface and its features more appealing to the general public and therefore they suggested a lot of help features and attractive features to get the interest of external users. Overall, the comments showed that the prototype had the potential to become a widely used raster data management system.

Improvements were made to the prototype following the evaluation. The time-series image viewing user interface was modified to allow users to view the same region of interest when they use the "next image" or "previous image" facility. This was achieved by passing a bounding box, or region extent, to the image viewing client when it loads the next or previous images in the time-series.

On the image search form, terms were rearranged to bring forward more understandable terms such as "keyword" and "theme". It was also clearly indicated that all search terms were optional and were only meant to refine the search criteria.

On the same form, names of data types available for search were changed such that they made more sense to users both within and outside the RRSU. For example, *"CPC_RFE_SADC"* changed to "SADC-wide Rainfall Estimates", and *"NOAA_NDVI_Africa"* changed to "Africa-wide Vegetation Map".

Chapter 8 — Conclusion

8.1 Summary

This thesis presents applied research in raster data management in the context of early warning for food security. It was mainly motivated by the availability of a wide range of open source software which offers the opportunity to develop customised solutions for geographical data management. This research proposed a system architecture that offers web-based data discovery and analysis tools adapted to time-series raster datasets. This approach handles the problem of time-series raster datasets discovery by offering a combination of a metadatabase management system, an image search facility and a web map service, which addresses the needs of both the users and managers of the raster data. A prototype implementation of the architecture was built for the RRSU. The prototype used PostgresSQL DBMS, with spatial functions offered by PostGIS, Java MapScript, Java Advanced Imaging API, MapBuilder, Java Servlets and Java Server Pages. Because the system was developed using open source and free software, it can be accessed easily and customised for other uses.

The proposed architecture is a 3-layered system with the following components:

- A metadatabase and a prescribed metadata schema designed to effectively represent time-series raster data collections.
- A web map server for metadata and image retrieval and display, with image analysis functionality for common tasks such as the comparison of images with their reference images.
- A user interface for experienced and novice data managers and data users.

The main components are summarised in the following sections.

8.1.1 Metadata Schema

A metadata schema was developed to adequately document information about time-series raster datasets, using terminology modified to a vocabulary familiar to targeted users. The metadatabase schema separates static metadata, which apply to an entire time-series set, from dynamic (or temporal) metadata, and thus significantly reduces the burden of raster metadata documentation since the static elements can be entered once only. This is particularly suited to the task of documenting metadata for time-series raster datasets which share a lot of common static attributes.

8.1.2 Metadata Management

A metadatabase management system with a user-friendly interface is required for data managers to adequately provide a standard description of time series raster datasets.
This interface should save time for raster data managers by allowing them to provide only the temporal metadata of raster datasets which are part of a time series. The remaining (or static) metadata, pertaining to the time series collection as a whole, is provided once only before metadata for the individual raster datasets is added.

8.1.3 Raster Data Discovery

A search facility which interacts with the metadatabase facilitates raster data discovery. This metadata allows the user to learn more about the raster data and to retrieve it from the source specified by the metadata. An overview of the raster data (like a thumbnail) allows the user to view the raster data in a dedicated web map service.

8.1.4 Web Map Service

The web map service completes the raster data discovery solution by allowing users to view and interrogate the raster data. Raster datasets can be viewed with other geographical datasets such as administrative boundaries to help the users understand and interpret the images. Users can query the raster data and receive thematic information. The web map service allows users to navigate through a time-series set of raster datasets by simply pressing a "next image" or "previous image" button, and is also used to display the results of raster data analyses.

8.1.5 Raster Data Analysis

The provision of some basic raster or image data analysis utilities assists users by automating some routine processing. These functions were chosen based on observations of the main tasks carried out by the target users and also on interviews with the target users.
These functions are: (i) averaging of raster datasets over a given time span (ii) summing up of raster datasets (iii) comparing sums of raster datasets versus reference raster datasets.
Reusable Java classes were developed to handle the raster data analysis functions. While the functions were primarily meant for users in raster-data based food security analysis, they can also be useful in environmental applications such as hydrological modelling, and other areas including economics.

8.1.6 Usability and Acceptance

Experiments conducted during the evaluation of the prototype system showed that the web-based prototype facilitates raster data discovery and display without the use of specialised software tools. Users who participated in the evaluation were able to carry out the desired tasks with ease and in less time than they would do using the software packages that they traditionally used. Using the prototype, users saved from 60 to 85 % of their time when performing routine image data management and analysis tasks, with more time savings for the more complex tasks. The ease of use and significant time savings were the main factors in the acceptance of the prototype by the targeted department. The targeted users were able to identify ways in which the prototype could contribute to their work routines. The prototype system is already being used to document time-series raster datasets and to perform basic discovery and analysis functions.

Overall, this research produced a metadatabase design for time-series raster data, coupled with an open source web-based system architecture for fast, comprehensive and reliable entry and searching of metadata, including immediate viewing and analysis of images.

8.2 Future Work

As noted during the evaluation of the prototype, the prototype had some problems and limitations. The main improvements can be made to the prototype in the following ways.

8.2.1 User logins and session management

During the interactions with the users, some felt that the database management functions which allow altering of the database should be available only to certain "super users". This would call for a facility to manage users and perform some authentication of user details and allow logins. A means for user session tracking would also need to be implemented. This could be done using the *HttpSession* interface provided by the Java Servlet specification.

8.2.2 Polygon-based extraction of data

Polygon-based extraction of data from the images discovered via the prototype system was not addressed. As shown in the introduction section of this thesis, the users sometimes perform data extraction from the raster data for areas of interest and this data is used to plot time-series graphs used in agricultural season monitoring. The extraction of information from the images could also be done using user defined areas of interest. This requires a mechanism for users to provide their region of interest either by drawing on the screen or providing files that define their areas of interest. The prototype that was implemented only offers point-based information extraction. The OpenGIS web feature service (WFS) standard could be investigated and implemented as part of this research.

8.2.3 Content-based raster data searches

Related to information extraction outlined above, some more work could be done to enable raster data searches based on the content of the raster datasets, for example, searching for raster data which show below average conditions (e.g. below average vegetation performance). This facility could assist users to easily discover anomalous information in the raster datasets.

8.2.4 Data Downloading

The prototype allows users to download image data as is. Evaluators indicated that the downloading facility is inconvenient as it does not allow the users to choose the format of the data to be downloaded and portion of the image, or region of interest, to be downloaded. This need can be handled by the implementation of a Web coverage service (WCS) [7, 32]. The new facility needs to handle the image formats available in the metadatabase and be able to convert to common formats requested by users.

8.2.5 Interface Usability Improvement

In the evaluation of the prototype user interface, users asked for flexibility in the use of colour schemes for the display of images. The prototype only offered the use of one colour scheme for each image type, but users indicated they may want to use colours that are different from the ones commonly used, for example, if they wish to highlight certain ranges of values in the image by using a more prominent colour.

This feature requires significant modification of both the user interface and the web map server. The user interface needs to be modified to give users a means to specify which colour schemes they want to use. An appropriate mechanism for passing this information to the web map server needs to be implemented so that the user interface receives a refreshed image with the desired colours.

Users also indicated that the image search facility could be improved by the addition of a facility to allow the specification of region of interest using a map facility rather than the "West-East-North-South" coordinate bounds used on the image search form (see Figure 8.1).

To address this need, an interactive map panel needs to be added to the form to capture coordinates when users draw out a region of interest of the map, and these coordinates will be used to refine the image search criteria.

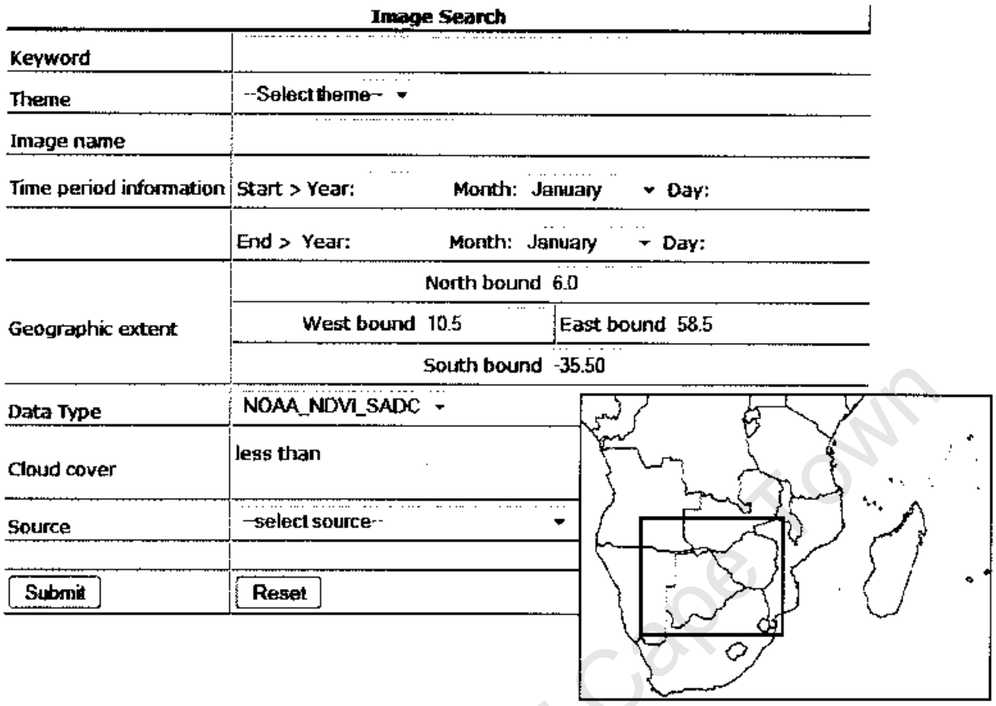


Figure 8.1: Image Search form with suggested improvements

References

[1] AHONEN-RAINIO, P, 2005. *Visualization Of Geospatial Metadata For Selecting Geographic Datasets.* ISBN 951-22-7525-2. Helsinki University of Technology. Available: http://lib.tkk.fi/Diss/2005/isbn9512275252/isbn9512275252.pdf [8 December 2007]

[2] AHONEN-RAINIO, P. *Concept Testing of Some Visualization Methods for Geographic Metadata.* Helsinki University of Technology. Available: http://www.scangis.org/scangis2003/papers/32.pdf [8 December 2008]

[3] APACHE SOFTWARE FOUNDATION. *Apache Tomcat.* Available: http://tomcat.apache.org/index.html. [February 20 2008]

[4] BAUMANN, P, et al. *Spatio-Temporal Retrieval with RasDaMan.* Proceedings of the 25th VLDB Conference, Edinburgh, Scotland, 1999.

[5] BAUMANN, P. *Web-enabled Raster GIS Services for Large Image and Map Databases.* 5th Intl Workshop on Query Processing and Multimedia Issues in Distributed Systems (QPMIDS'2001)

[6] BAUMANN, P. *Large-Scale, Standards-Based Earth Observation Imagery and Web Mapping Services.* Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003.

[7] BAUMANN, P. *Towards a Standard for Interoperable Earth System Raster Services.* Shaping the Change XXIII FIG Congress, 2006

[8] BENEDICT, K. K., MENKE, K., HUDSPETH, B., CAVNER, J. (2005). *EDAC's Web-based Geospatial Applications and the Open Source Technologies Behind Them.* Albuquerque: University of New Mexico

[9] CALSAVARA, A. P. (2002). *A portable alternative for BLOBs.* Available: http://www.builderau.com.au/architect/database/soa/A_portable_alternative_for_**BLOB**s/0,39024547,20267938,00.htm. [5 August 2007]

[10] CHARLIER, I. (1999). *WINDISP 4.0 User's Manual, Multilingual Version.* Rome: FAO Global Information and Early Warning System.

[11] CHENG P, LI, J., LIU, C., LIAO, T. Study and Design of Geo-spatial metadata managing system based on USDI. In *ISPRS Workshop on Service and Application of Spatial Data Infrastructure, XXXVI(4/W6),* Hangzhou, China. 2005

[12] CODEHAUS FOUNDATION, 2006. GeoTools, The Open Source Java GIS Toolkit. Available: http://geotools.codehaus.org/ [15 August 2007]

[13] CODEHAUS FOUNDATION (2006). *GeoTools Developers Guide.* Available: http://www.geotools.org/ [15 August 2007]

[14] DENG, Y. (2002). The Metadata Architecture for Data Management in Web-based Choropleth Maps

[15] DOMIK, G. (1996). *Tutorial on Visualization. University of Paderborn.* Available: http://wwwcs.uni-paderborn.de/cs/vis/hypervis/domik/folien.html (20 September 2007)

[16] DUBLIN CORE METADATA INITIATIVE (DCMI) (2006). *Dublin Core Metadata Element Set, Version 1.1.* Available: http://dublincore.org/documents/dces/ [20 September 2007]

[17] FAO (FOOD AND AGRICULTURE ORGANIZATION) (2007). *GIEWS Workstation.* Retrieved September 12 2007 from FAO Food Security for Decision Making: http://www.foodsec.org/tools_ew_01.htm.

[18] FEDERAL GEOGRAPHIC DATA COMMITTEE (1998). *FGDC-STD-001-1998. Content standard for digital geospatial metadata (revised June 1998).* Washington D. C.: Federal Geographic Data Committee.

[19] GARRETT J. J. (2005). *Ajax: A New Approach to Web Applications.* Available: http://adaptivepath.com/publications/essays/archives/000385.php [3 May 2007]

[20] GATTASS, M., FERREIRA, C. C. F., VILAR, A. S., GLASBERG, M. S. (1999). *Efficient Map Visualization on the Web.* Rio de Janeiro: Pontifical Catholic University of Rio de Janeiro. Available: http://www.tecgraf.puc-rio.br/publications/artigo_1999_efficient_map_visualization.pdf

[21] HAAS, W. (2007). *Database Shootout: Benchmarking spatial DBMSs.* FOSS4G2007, Victoria, Canada.

[22] HAECHLER, T., 2003. *Online Visualization of Spatial Data. A Prototype of an open source Internet Map Server with backend spatial database for the Swiss National Park.* Zurich: University of Zurich.

[23] HASTINGS D. A. (1997). *The Geographic Information Systems: GRASS HOWTO* Available: **http://www.tldp.org/HOWTO/GIS_GRASS/index.html** [3 May 2007]

[24] HUTCHINSON, C. F. (1998). *Social Science and Remote Sensing in Famine Early Warning.* **In** Committee on the Human Dimensions of Global Change, National Research Council. *People and Pixels: Linking Remote Sensing and Social Science.* The National Academies Press.

[25] ISO (2003). *ISO 19115, Geographic information — Metadata.* Reference number ISO 19115:2003(E)

[26] LAT/LON GMBH (2006). *deegree Web Coverage Service.* Bonn University. Available: http://www.webmap-test.thueringen.de/deegreewms/doc/deegree_WCS_configuration_2003-09-12.pdf

[27] LILLESAND, T.M. & KIEFER, R.W. (2000). *Remote sensing and image interpretation, 4th Edition.* John Wiley, New York.

[28] MING-HSIANG, T. (2005). Recent developments in Internet GIS. Available: http://www.gisdevelopment.netAechnology/gis/techgis 002pfhtm [10 April 2007]

[29] MYSQL AB. MySQL. http://www.mysql.com/, 2007.

[30] NATIONAL INFORMATION STANDARDS ORGANIZATION (NISO) (2004). *Understanding Metadata.* NISO Press, USA, ISBN: 1-880124-62-9.

[31] NEBIKER, S. *Spatial Raster Data Management for Geo-Information Systems — A Database Perspective.* Zurich: ETH Zurich, 1997.

[32] OPEN GIS CONSORTIUM (OGC) Inc. (2003). *Web Coverage Service (WCS), Version 1.0.0.* Available: http://www.opengeospatial.org/standards/wcs

[33] OPEN GIS CONSORTIUM (OGC) Inc. (2006). *Web Map Server Implementation Specification.* Available: http://www.opengeospatial.org/standards/wms

[34] ORACLE (2006). *Oracle Database 10g: Managing Spatial Raster Data Using GeoRaster. An Oracle Technical Whitepaper.* Available: http://www.pci.on.ca/pdfs/Geomatica%200racle%20Whitepaper.pdf

[35] OTEY, M. (2002). *Return of the BLOB.* SQL Server Magazine. Available: http://msdn2.microsoft.com/en-us/library/aa496014%28SOL.80%29.aspx

[36] PCI GEOMATICS (2006). *Geomatica Image Management System (GIMS). A PCI Geomatics White Paper. Available:* *http://www.pci.on.ca/solutions/gims/pdfs/GIMS Whitepaper.pdf* [15 August 2007]

[37] PHILLIPS, A. H. *A Metadata Management System for Web Based SDIs.* Victoria: University of Melbourne,1998.

[38] PostGIS. http://postgis.refractions.net/

[39] REGENTS OF THE UNIVERSITY OF MINNESOTA (2008). *MapServer.* Available: http://mapserver.gis.umn.edu [12 February 2008]

[40] SAYAR, A., PIERCE, M., FOX, G., (2005a). *Developing GIS Visualization Web Services For Geophysical Applications.* Indiana University. Available: http://grids.ucs.indiana.edu/ptliupages/publications/isprs asayar.pdf [15 April 2007]

[41] SCOTT M. Lewandowski, 1998. *Frameworks for component-based client/server computing,* ACM Computing Surveys (CSUR), v.30 n.1, p.3-27, March 1998

[42] SHAIG, A. An Overview of Web based Geographic Information Systems. In *The 13th Annual Colloquium of the Spatial Information Research Centre,* University of Otago, Dunedin, New Zealand, December 2nd-5th 2001.

[43] SUN MICROSYSTEMS, INC. (1999). *Programming in Java Advanced Imaging Release 1.0.1*. California. Available: http://d/c.sun.com/pdf/806-5413-10/806-5413-10.pdf [15 May 2007]

[44] SUN MICROSYSTEMS, INC. (2004). *JDK 5.0 Documentation.* Available: http://java.sun.com/j2se/1.5.0/docs/ [16 May 2007]

[45] SUN MICROSYSTEMS, INC. (200/). *Java 2DTM API. Programmer's Guide to the JavaTM 2D API.* Available: http://java.sun.com/j2se/1.5.0/docs/guide/2d/spec/j2d-bookTOC.html [25 May 2007]

[46] SUN MICROSYSTEMS, INC. (2001). *Java Image I/O API. Programmer's Guide to the JavaTM 2D API.* Available: http://java.sun.com/javase/6/docs/technotes/guides/imageio/index.html [22 May 2007]

[47] SUN MICROSYSTEMS, INC. (2007). *JAI What is Java Advanced Imaging.* Available: http://java.sun.com/products/java_media/jai/whatis.html [25 May 2007]

[48] SUN MICROSYSTEMS, INC (2003). BufferedImage. http://java.sun.com/j2se/1.4.2/docs/api/java/awt/image/BufferedImage.html

[49] SUN MICROSYSTEMS, INC (2003). RenderedImage. http://java.sun.corn/j2se/1.4.2/docs/api/java/awt/image/RenderedImage.html

[50] SUN MICROSYSTEMS, INC. Tiledlmage. http://java.sun.com/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/jai/TiledImage.html

[51] TICHELER, J. (2005). *GeoNetwork opensource Spatial Data Catalog, A Common Spatial Data Management Strategy*. Open Source Geospatial 2005, Minneapolis

[52] TU, S., FLANAGIN, M., WU, Y., ABDELGUERFI, M., NORMAND, E., MAHADEVAN, V., (2004). *Design Strategies to Improve Performance of GIS Web Services.* New Orleans: University of New Orleans. Available: http://dmap.nr/ssc.navy.mil/dmap/pubs/documents/itcc2004.pdf [12 April 2007]

[53] VAN WIJK, J. J., 2005. *The Value of Visualization.* IEEE Visualization 2005. IEEE Visualization 2005.

[54] W3C (2003). *Scalable Vector Graphics (SVG) 1.1 Specification.* Available: http://www.w3.org/TR/SVG11/

[55] W3C (2001). *WebCGM 1.0 Second Release.* Available: http://www.w3.org/TR/REC-WebCGM/ [20 September 2007]

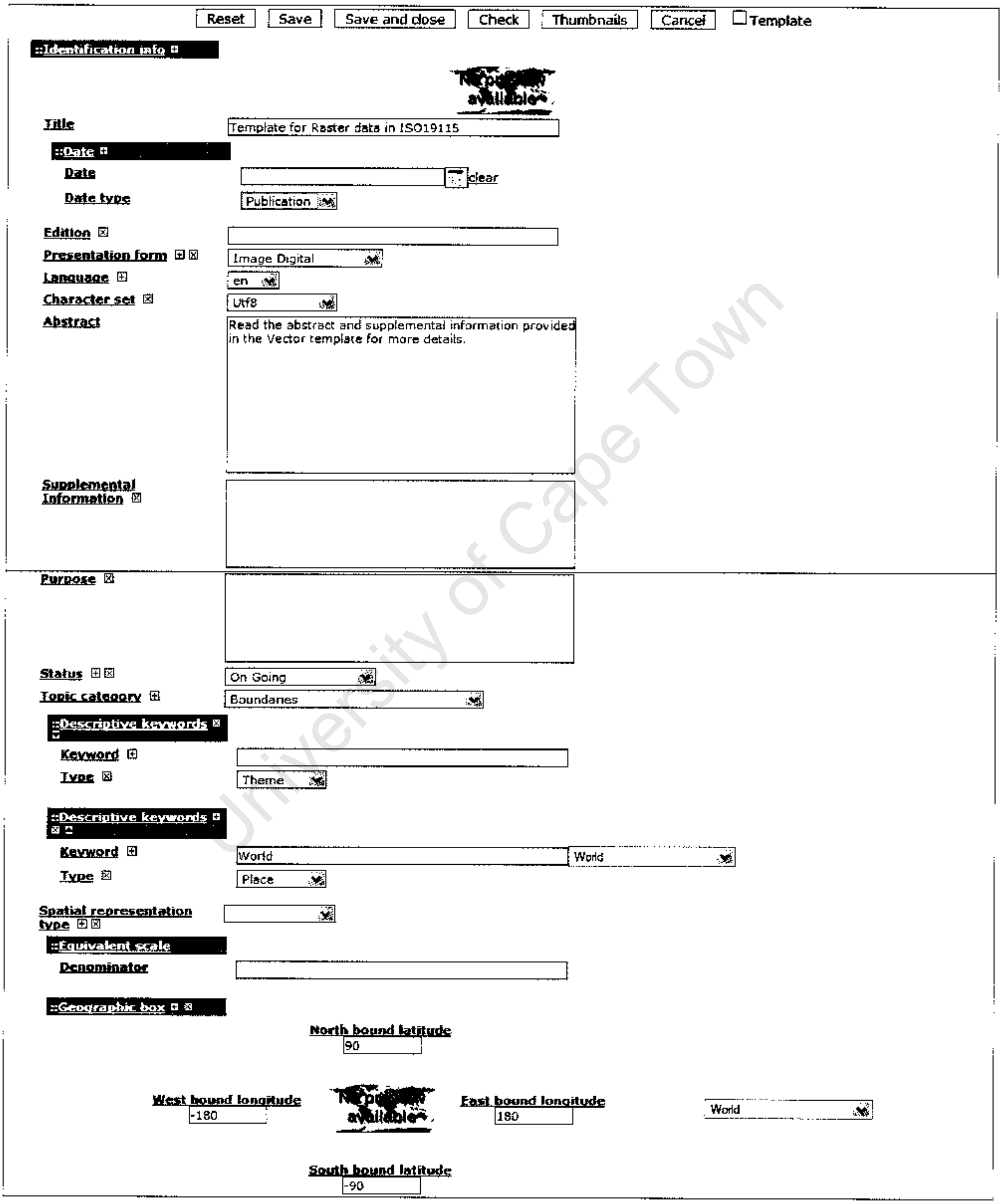
[56] WORLD METEOROLOGICAL ORGANIZATION (1992). *International Meteorological Vocabulary*. WMO Publication 182, Second Edition, Geneva, Switzerland 784 p.

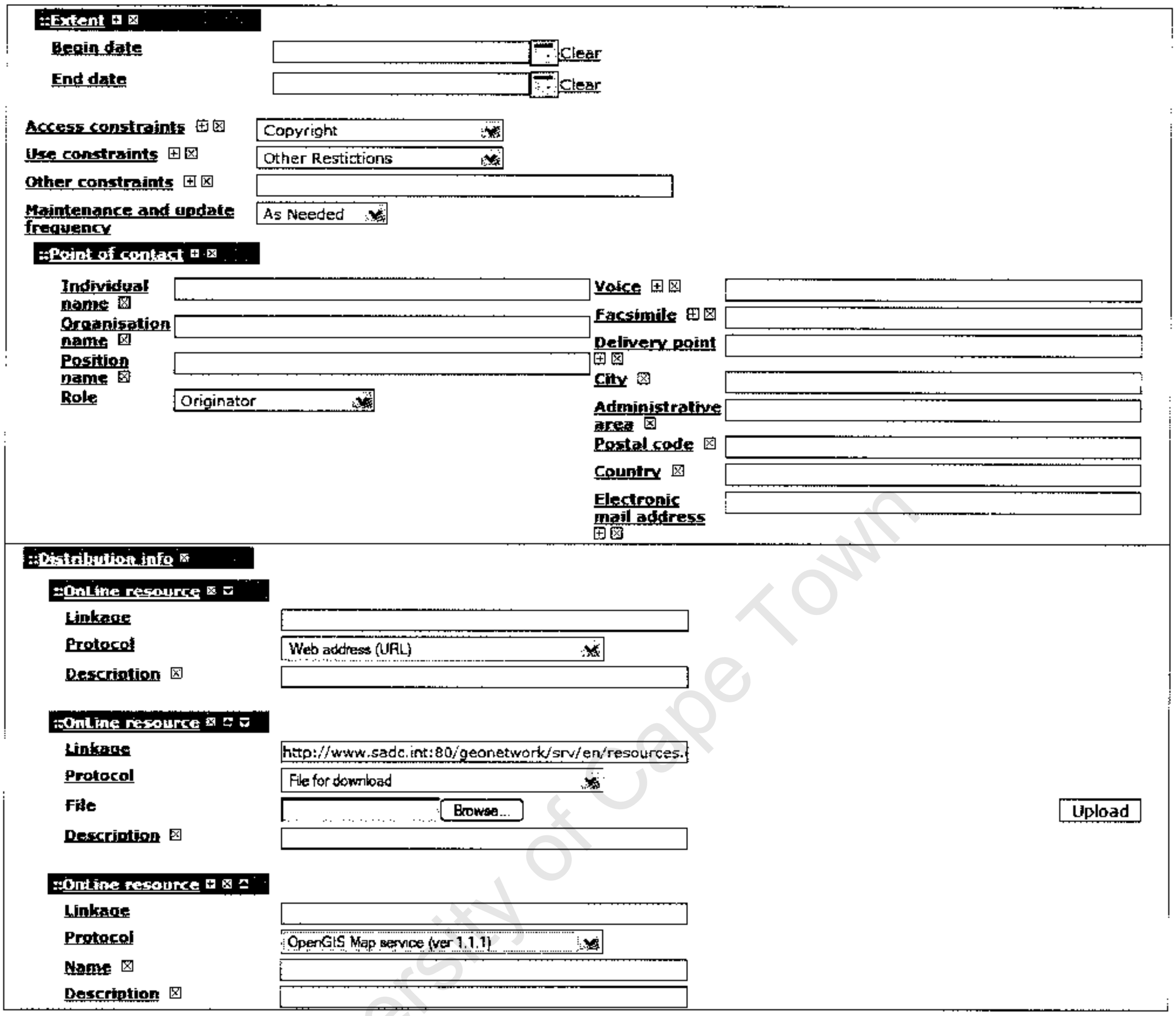
[57] WOJNAROWSKA, M., ADY, B. E., 2002. *Interoperable Solutions in Web-based Mapping.* Symposium on Geospatial Theory, Processing and Applications, Ottawa 2002.

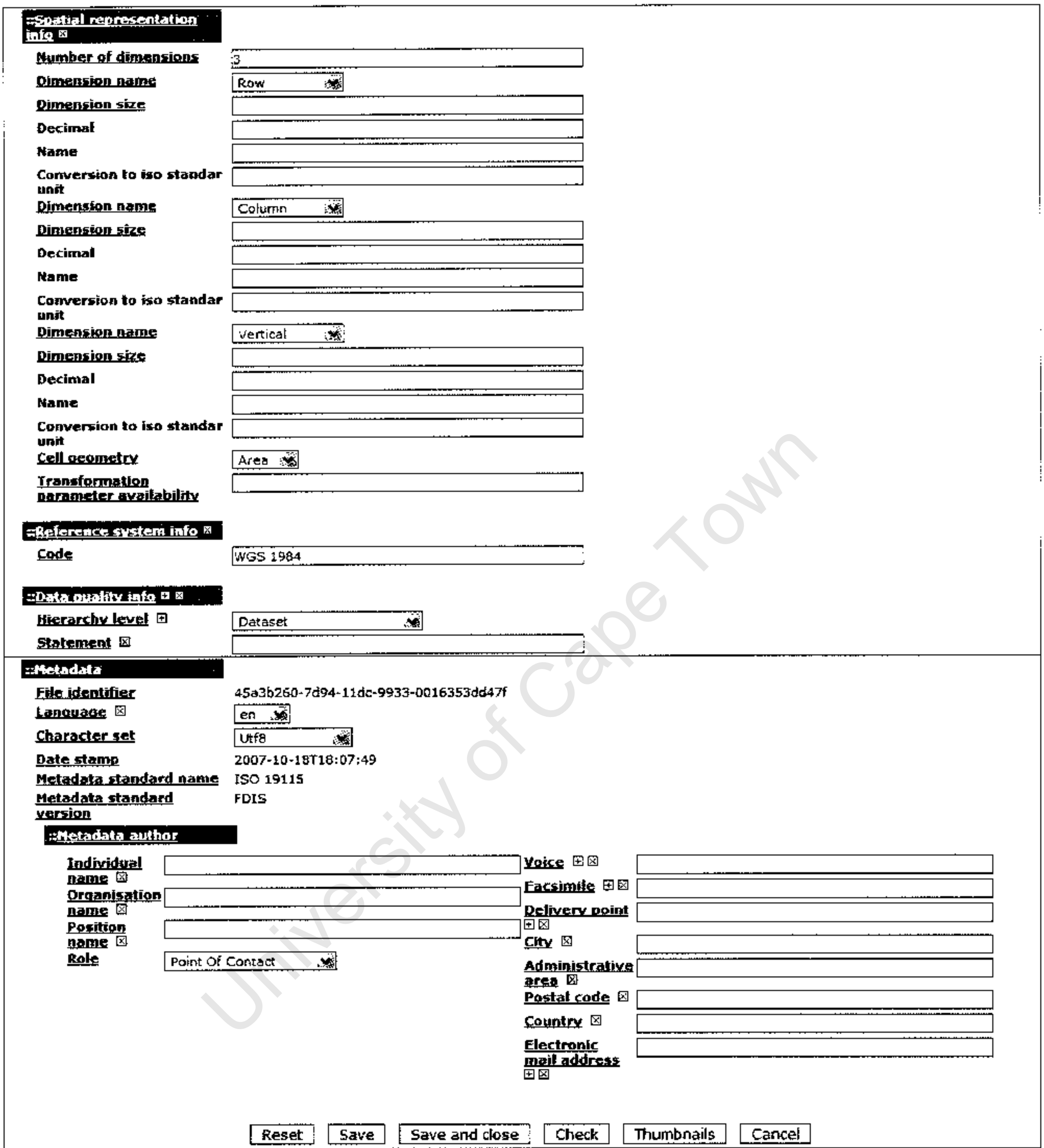
[58] Wu et al. *The CenSSIS Image Database.* Center for Subsurface Sensing and Imaging Systems. Northeastern University, Boston. (2003)

[59] YURTSEVEN, L. (2006). *Quick Start Guide — GeoNetwork opensource Version 2.0.* New York: United Nations - OCHA - Office for the Coordination of Humanitarian Affairs.

Appendix A - FAO GeoNetwork template for ISO 19115 Raster metadata




::Extent
Begin date
Clear
End date
Clear
Access constraints
Copyright
Use constraints
Other Restictions
Other constraints
Maintenance and update frequency
As Needed
::Point of contact
Individual name
Organisation name
Position name
Role
Originator
Voice
Facsimile
Delivery point
City
Administrative area
Postal code
Country
Electronic mail address
::Distribution info
::OnLine resource
Linkage
Protocol
Web address (URL)
Description
::OnLine resource
Linkage
http://www.sadc.int:80/geonetwork/srv/en/resources.
Protocol
File for download
File
Browse...
Upload
Description
::OnLine resource
Linkage
Protocol
OpenGIS Map service (ver 1.1.1)
Name
Description



::Spatial representation info

Number of dimensions: 3

Dimension name: Row

Dimension size:

Decimal:

Name:

Conversion to iso standar unit:

Dimension name: Column

Dimension size:

Decimal:

Name:

Conversion to iso standar unit:

Dimension name: vertical

Dimension size:

Decimal:

Name:

Conversion to iso standar unit:

Cell geometry: Area

Transformation parameter availability:

::Reference system info

Code: WGS 1984

::Data quality info

Hierarchy level: Dataset

Statement:

::Metadata

File identifier: 45a3b260-7d94-11dc-9933-0016353dd47f

Language: en

Character set: Utf8

Date stamp: 2007-10-18T18:07:49

Metadata standard name: ISO 19115

Metadata standard version: FDIS

::Metadata author

Individual name:

Organisation name:

Position name:

Role: Point Of Contact

Voice:

Facsimile:

Delivery point:

City:

Administrative area:

Postal code:

Country:

Electronic mail address:

Reset | Save | Save and close | Check | Thumbnails | Cancel

Source: www.sadc.int/geonetwork

Appendix B - FGDC CSDGM elements

Identification_Information
Citation +
Description + (Abstract + Purpose + (Supplemental_Information)
Time_Period_of_Content + Time_Period_Information + Currentness_Reference
Status + (Progress + Maintenance_and_Update_Frequency)
Spatial_Domain + Bounding_Coordinates + (1{Data_Set_G-Polygon}n - coordinates defining the outline of an area covered by a data set)
Keywords + (1{Theme}n + 0{Place}n + 0{Stratum}n + 0{Temporal}n)
Access_Constraints +
Use_Constraints +
(Point_of_Contact) +
(1{Browse_Graphic = *Browse_Graphic_File_Name + Browse_Graphic_File_Description + Browse_Graphic_File_Type*)}n) +
(Data_Set_Credit) +
(Security_Information) + *Security_Classification_System + Security_Classification + Security_Handling_Description*
(Native_Data_Set_Environment) +
(1{Cross_Reference}n)

Data_Quality_Information
Data_Quality_Information *0{Attribute_Accuracy}1 + Logical_Consistency_Report + Completeness_Report + Positional_Accuracy 0-1 + Lineage + (Cloud_Cover) + Attribute_Accuracy = (Attribute_Accuracy_Report + (1{Quantitative_Attribute_Accuracy_Assessment}n)*
Quantitative_Attribute_Accuracy_Assessment = *Attribute_Accuracy_Value + Attribute_Accuracy_Explanation*
Positional_Accuracy = *0{Horizontal_Positional_Accuracy}1 + 0{Vertical_Positional_Accuracy}1*
Horizontal_Positional_Accuracy = *Horizontal_Positional_Accuracy_Report + (1{Quantitative_Horizontal_Positional_Accuracy_Assessment}n)*
Quantitative_Horizontal_Positional_Accuracy_Assessment = *(Horizontal_Positional_Accuracy_Value + Horizontal_Positional_Accuracy_Explanation*
Vertical_Positional_Accuracy = *(Vertical_Positional_Accuracy_Report + (1{Quantitative_Vertical_Positional_Accuracy_Assessment}n)*
Quantitative_Vertical_Positional_Accuracy_Assessment = *Vertical_Positional_Accuracy_Value + Vertical_Positional_Accuracy_Explanation*
Lineage = *(0{Source_Information}n + 1{Process_Step}n*
Source_Information = *Source_Citation + 0{Source_Scale_Denominator}1 + Type_of_Source_Media + Source_Time_Period_of_Content + Source_Citation_Abbreviation + Source_Contribution*
Source_Citation = *Citation_Information*
Source_Time_Period_of_Content = *Time_Period_Information + Source_Currentness_Reference*
Process_Step = *Process_Description + 0{Source_Used_Citation_Abbreviation}n + Process_Date + (Process_Time) + 0{Source_Produced_Citation_Abbreviation}n + (Process_Contact)*
Process_Contact = *Contact_Information*

Spatial_Data_Organization_Information
Spatial_Data_Organization_Information = *(0{Indirect_Spatial_Reference}1 + 0{Direct_Spatial_Reference_Method + ({Point_and_Vector_Object_Information + Raster_Object_Information})}1*
Raster_Object_Information = *Raster_Object_Type + (Row_Count + Column_Count + 0{Vertical_Count}1)*

Spatial_Reference_Information

Spatial_Reference_Information = *0{Horizontal_Coordinate_System_Definition}1 + 0{Vertical_Coordinate_System_Definition}1*
Horizontal_Coordinate_System_Definition = *[Geographic | 1{Planar}n | Local] + 0{Geodetic_Model}1*
Geographic = *Latitude_Resolution + Longitude_Resolution + Geographic_Coordinate_Units*
Planar = *[Map_Projection | Grid_Coordinate_System | Local_Planar] + Planar_Coordinate_Information*

Entity_and_Attribute_Information
Entity_and_Attribute_Information = *[1{Detailed_Description}n | 1{Overview_Description}n | 1{Detailed_Description}n + 1{Overview_Description}n]*
Detailed_Description = *Entity_Type + 0{Attribute}n*
Entity_Type = *Entity_Type_Label + Entity_Type_Definition + Entity_Type_Definition_Source*
Attribute = *Attribute_Label + Attribute_Definition + Attribute_Definition_Source + 1{Attribute_Domain_Values}n + 0{Beginning_Date_of_Attribute_Values + 0{Ending_Date_of_Attribute_Values}1}n + (Attribute_Value_Accuracy_Information) + (Attribute_Measurement_Frequency)*
Attribute_Domain_Values = *[Enumerated_Domain | Range_Domain | Codeset_Domain | Unrepresentable_Domain]*
Enumerated_Domain = *1{Enumerated_Domain_Value + Enumerated_Domain_Value_Definition + Enumerated_Domain_Value_Definition_Source + 0{Attribute}n }n*
Range_Domain = *Range_Domain_Minimum + Range_Domain_Maximum + 0{Attribute_Units_of_Measure}1 + (Attribute_Measurement_Resolution) + 0{Attribute}n*
Codeset_Domain= *Codeset_Name + Codeset_Source*
Attribute_Value_Accuracy_Information = *Attribute_Value_Accuracy + Attribute_Value_Accuracy_Explanation*
Overview_Description = *Entity_and_Attribute_Overview + 1{Entity_and_Attribute_Detail_Citation}n*

Distribution_Information
Distribution_Information = *Distributor + 0{Resource_Description}1 + Distribution_Liability + 0{Standard_Order_Process}n + 0{Custom_Order_Process}1 + (Technical_Prerequisites) + (Available_Time_Period)*
Distributor = *Contact_Information*
Standard_Order_Process = *[Non-digital_Form | 1{Digital_Form}n] + Fees + (Ordering_Instructions) + (Turnaround)*
Digital_Form = *Digital_Transfer_Information + Digital_Transfer_Option*
Digital_Transfer_Information = *Format_Name + ([Format_Version_Number | Format_Version_Date] + (Format_Specification)) + (Format_Information_Content) + 0{File_Decompression_Technique}1 + (Transfer_Size)*
Digital_Transfer_Option = *1{ [Online_Option | Offline_Option] }n*
Online_Option = *1{Computer_Contact_Information}n + (Access_Instructions) + (Online_Computer_and_Operating_System)*
Computer_Contact_Information = *[Network_Address | Dialup_Instructions]*
Network_Address = *1{Network_Resource_Name}n*
Dialup_Instructions = *Lowest_BPS + 0{Highest_BPS}1 + Number_DataBits + Number_StopBits + Parity + 0{Compression_Support}1 + 1{Dialup_Telephone}n + 1{Dialup_File_Name}n*
Offline_Option = *Offline_Media + 0{Recording_Capacity}1 1{Recording_Format}n + 0{Compatibility_Information}1*
Recording_Capacity = *1{Recording Density}n + Recording_Density_Units*
Available_Time_Period = *Time_Period_Information*

Metadata_Reference_Information
Metadata_Reference_Information = *Metadata_Date + (Metadata_Review_Date) + (Metadata_Future_Review_Date) + Metadata_Contact + Metadata_Standard_Name +*

Metadata_Standard_Version + 0{Metadata_Time_Convention}1 + (Metadata_Access_Constraints) + (Metadata_Use_Constraints) + (Metadata_Security_Information) + 0{Metadata_Extensions}n
Metadata_Contact = *Contact_Information*
Metadata_Security_Information = *Metadata_Security_Classification_System + Metadata_Security_Classification + Metadata_Security_Handling_Description*
Metadata_Extensions = *0{Online_Linkage}n + 0{Profile_Name}1*

Citation_Information
Citation_Information = *1{Originator}n + Publication_Date + (Publication_Time) + Title + 0{Edition}1 + 0{Geospatial_Data_Presentation_Form}1 + 0{Series_Information}1 + 0{Publication_Information}1 +*
0{Other_Citation_Details}1 + (1{Online_Linkage}n) + 0{Larger_Work_Citation}1
Series_Information = *Series_Name + Issue_Identification*
Publication_Information = *Publication_Place + Publisher*
Larger_Work_Citation = *Citation_Information*

Time_Period_Information
Time_Period_Information = *[Single_Date/Time | Multiple_Dates/Times | Range_of_Dates/Times]*
Single_Date/Time = *Calendar_Date + (Time_of_Day)*
Multiple_Dates/Times = *2{Single_Date/Time}n*
Range_of_Dates/Times = *Beginning_Date + (Beginning_Time) + Ending_Date + (Ending_Time)*

Contact_Information
Contact_Information = *[Contact_Person_Primary | Contact_Organization_Primary] + (Contact_Position) + 1{Contact_Address}n + 1{Contact_Voice_Telephone}n + (1{Contact_TDD/TTY_Telephone}n) + (1{Contact_Facsimile_Telephone}n) + (1{Contact_Electronic_Mail_Address}n) + (Hours_of_Service) + (Contact_Instructions)*
Contact_Person_Primary = *Contact_Person + (Contact_Organization)*
Contact_Organization_Primary = *Contact_Organization + (Contact_Person)*
Contact_Address = *Address_Type + 0{Address}n + City + State_or_Province + Postal_Code + (Country)*

Symbols used above
(..element…) – element is optional
[a | b | c] – choice between a, b and c
0{..element..}1 – mandatory if applicable
m{..element..}n – element is repeated m to n times

Appendix C - User consultation in the determination of required metadata

M1

Information to be captured by **metadata elements** in a **metadatabase**
- to allow efficient image discovery and appropriate use

	Required for storage	**Required for display**
Title ~~/ name of file~~	✓	✓
Purpose of the image		✓
Theme		✓
Time-series (boolean – true or false, if yes which time-series, what time step/interval, existing or new)	✓	
Parameter (e.g vgt NDVI) / (this parameter will allow image to be related to other images of representing the same parameter)		✓
Valid value range for the parameter being measured (eg 0.1-0.8 for NDVI / 0-253 for rainfall)	[illegible]	✓
Radiometric resolution [8-bit, 16-bit]		✓
		[illegible]
Spatial resolution of the image		✓
Geographic projection [have predefined list of common projections]		✓
Reference ellipsoid		✓
Geographic extent (minimum bounding rectangle, bounding polygon for images)		✓
Image size (width x height)		✓
Image format (n bytes, binary, etc) → Is a new "floating point"?		✓
Number of bands in the image		✓
Byte layout (plain binary, BIL, BSP)		✓
Date of acquisition / entry into the system		✓
Time of validity period represented by the parameter / time period of content		✓
Data quality information (positional accuracy, cloud contamination, etc)		✓
File size (for advice on possible processing or download time)		✓
URL or file location for the image		✓
Compression (nil / Winzip / Winrar / password)		✓
Distribution information – mode of delivery, format, etc		✓
Usage constraints		✓
Source and alternative sources of image (e.g. ADDS)		
Legend (colour map or file)		
Decoder software list (WinDisp, etc)		

linked

Comments:

Information to be captured by **metadata elements** in a **metadatabase** M2
- to allow efficient image discovery and appropriate use

	Required for storage	Required for display
Title / name of file	✓	✓
Purpose of the image	✓	✓
Theme	✓	
Time-series (boolean – true or false, if yes which time-series, what time step/interval, existing or new) ~~User should create own~~ = specify range		✓
Parameter (e.g vgt NDVI) / (this parameter will allow image to be related to other images of representing the same parameter)		✓
Valid value range for the parameter being measured (eg 0.1-0.8 for NDVI / 0-253 for rainfall)		✓
Radiometric resolution [8-bit, 16-bit]	✓	
Spatial resolution of the image		✓
Geographic projection [have predefined list of common projections]		✓
Reference ellipsoid	✓	✓
Geographic extent (minimum bounding rectangle, bounding polygon for images)		✓
Image size (width x height)	✓	
Image format (n bytes, binary, etc)	✓	
Number of bands in the image	✓	
Byte layout (plain binary, BIL, BSP)		✓
Date of acquisition / entry into the system	✓	
Time of validity period represented by the parameter / time period of content		✓
Data quality information (positional accuracy, cloud contamination, etc)		✓
File size (for advice on possible processing or download time)		✓
URL or file location for the image		
Compression (nil / Winzip / Winrar / password)	✓	
Distribution information – mode of delivery, format, etc & cost		✓
Usage constraints		✓
Source and alternative sources of image (e.g. ADDS)	✓	
Legend (colour map or file) Could be included in the map itself / during loading graphic		
Decoder software list (WinDisp, etc)		✓

Comments:

- Formula required to convert from image values to the parameter values, should be included
- Add browse graphic

M3

Information to be captured by **metadata elements** in a **metadatabase**
- to allow efficient image discovery and appropriate use

	Required for storage	Required for display
Title / name of file	✓	✓
Purpose of the image		✓
Theme		✓
Time-series (boolean – true or false, if yes which time-series, what time step/interval, existing or new)	✓	
Parameter (e.g vgt NDVI) / (this parameter will allow image to be related to other images of representing the same parameter)		✓
Valid value range for the parameter being measured (eg 0.1-0.8 for NDVI / 0-253 for rainfall)		✓
Radiometric resolution [8-bit, 16-bit]	✓	
Spatial resolution of the image	✓	
Geographic projection [have predefined list of common projections]		✓
Reference ellipsoid	✓	
Geographic extent (minimum bounding rectangle, bounding polygon for images)	✓	✓
Image size (width x height)	✓	
Image format (n bytes, binary, etc)	✓	
Number of bands in the image	✓	
Byte layout (plain binary, BIL, BSP)	✓	
Date of acquisition / entry into the system	✓	
Time of validity period represented by the parameter / time period of content	✓	
Data quality information (positional accuracy, cloud contamination, etc)		✓
File size (for advice on possible processing or download time)	✓	
URL or file location for the image	✓	
Compression (nil / Winzip / Winrar / password)	✓	
Distribution information – mode of delivery, format, etc	✓	
Usage constraints	✓	
Source and alternative sources of image (e.g. ADDS)	✓	
Legend (colour map or file)		✓
Decoder software list (WinDisp, etc)		✓

Comments:

Appendix D - Data Dictionary

Image Metadatabase Schema Element Set

Metadata Element	**Short Name** [as in database]	**Description**	**ISO 19115** equivalent	**Data Type**	**Domain / possible values**	**Examples**
Section 1						
Theme table						
Theme id	theme_id	Identifier for the theme		Integer		
Theme name	theme_name	Name of main theme of the image data	tpCat	varchar(40)		Vegetation
Comments	theme_comments	General comments on the theme		text	Free text	
Section 2						
Time series table						
time series id	tseries_id	Identifier for the time series		Integer	Auto incremental integer	*[integer number]*
time series name	tseries_name	Arbitrary name for the time series		varchar(40)		"NOAA_AVHRR _Timeseries"
time series start	tseries_start	The start date of the time series		Date		1981-04-11
time series end	tseries_end	The final date or period in the time series		Date	Date or period or TBD if series is continuing	2007-10-21
time series interval	tseries_interval	The interval of the time series in days		Integer	10 (dekadal), 16, 30 (monthly)	10
Comments	tseries_comments	General comments on the time series, including information on		text	Free text	"this time series has missing data

		whether there are gaps in the series				for the period year 1988"
Section 3						
Data Type Table						
Data type ID	datatype_id	The identifier of the data		Integer	Auto incremental integer	*[integer number]*
Data type name	datatype_name	The name or title of the data		varchar(25)	List of available dataset types	"SADC_NOAA_NDVI"
Data content type	data_content	Type of information represented by an image value	contentTyp	varchar(40)	Rainfall, vegetation, etc	vegetation index
Data Purpose	data_purpose	Specific usage of the data	idPurp, specUsage	varchar(40)	Free text	"To monitor vegetation status"
Data abstract	data_abstract	Short write-up on the data	idAbs	text	Free text	"the data was prepared by NASA for the purposes of monitoring global vegetation condition; …"
Data theme ID	data_theme	Identity of the main theme of the data [refer to section 1]		Integer	From List of available themes	*[integer number]*
Keywords	data_keywords	Common word(s) used to describe the data	keyword	varchar(60)		NDVI, vegetation
Data time series ID	data_timeseries	ID of the time series which the data forms or is part of [refer to section 2]		Integer		*[integer number]*
Data Lower Limit	data_lolimit	The lower limiting value of the data	minVal	double		0.15 (for NDVI)
Data Upper Limit	data_uplimit	The upper limiting value of the data	maxVal	double		0.8 (for NDVI)

Status	data_status	The status of the data	idStatus	varchar(15)	Completed, Historical archive, Obsolete, Ongoing, Planned, Required, Under development	Completed
Citation	data_citation	Citation information for the data	idCitation	Text	Free text	"National Space Science Administration (NASA)"
Credit	data_credit	Persons or organisations that are contributors to the data	idCredit	Text	Free text	"Dr. Tucker"
Source ID	data_source	ID of the source of the data		Integer		*[integer number]*
Alternative Source	data_altsource	An alternative source for the data		Integer	see Source ID	*[integer number]*
Contact ID	data_contact	Identity of person and organization associated with the data	idPoC, userContactInfo	Integer		*[integer number corresponding to user]*
Maintenance Frequency	data_maintFreq	Frequency at which changes and updates are made to the dataset after its initial completion	maintFreq	varchar(30)	Free Text	Every ten days
Distribution ID	data_distribution	Identity of the distribution scheme for the data		Integer		*[integer number]*
Data language	data_language	The language used in the data	dataLang	varchar(10)		English
Usage constraints	data_usageCons	Limitations that apply in the usage of the data		Text	Free Text	"this data should not be used to monitor vegetation at very localised scales"
Data extent	data_extent	Polygon that defines the spatial extent of the image, [or upper-left, upper-right, lower-right,	dataExt	Geometry	Polygon	POLYGON((100 100, 200 100, 200 200, 100 200, 100

		lower-left]				100))
Comments	data_comments	General comments about the data, including aspects not covered by the elements above		Text	Free text	*[any text relevant to the usage of the data]*
Section 4						
Image type table						
Type id	imagetype_id	Identifier for the image type		Integer		*[integer number]*
Image type name	imagetype_name	Name for the image type		varchar(25)		"CPC_RFE_SADC"
spatial resolution	spatialres_x, spatialres_y	The size of the area on the ground represented by each pixel in the image, in the X and Y directions respectively	dataScale	Double		0.1 degrees, 8000m
image rows	image_rows	The height of the image in pixels		Integer		1100
image columns	image_columns	The width of the image in pixels	dimSize	Integer		1200
geographic projection ID	geoprj_id	An ID which identifies the geographic projection used by the image		Integer		4326 *[latitude longitude projection]*
longitude centre	lat_centre	A reference longitude used for georeferencing the image		Double		0.0 degrees
Latitude centre	long_centre	A reference latitude used for georeferencing the image		Double		-1.5 degrees
X centre	image_xcentre	The pixel column number (image coordinate in the X direction) which coincides with the longitude centre		Double		200.5
Y centre	image_ycentre	The pixel row number (image coordinate in the Y direction)		Double		150.2

		which coincides with the latitude centre				
Image Lower limit	image_lolimit	The lowest value in the image representing a valid physical value		Integer		0
Image Upper limit	image_uplimit	The highest value in the image representing a valid physical value		Integer		250
Image Missing value	image_missingv	Image value that indicates missing data		Integer		253
Image format ID	image_format	ID of the format of the image [refer to section 13]		Integer		*[integer number from database]*
Image slope	image_slope	The scale factor which has to be applied to the image value	sclFac	Double		0.004
Image intercept	image_intercept	The physical value corresponding to image value of zero	offset	Double		-0.1
Dataset ID	image_datatype	The ID of the dataset that the image type belongs to		Integer	integer number representing data type	*[integer number]*
Decoder ID	image_decoder	The ID of the software that can decode this image type		Integer	integer number representing decoder	*[integer number]*
Colour table ID	image_clrtable	The ID of the main colour table or legend used to display this image type		Integer	integer number representing colour table	*[integer number]*
Section 5						
Colour tables table						
colour table ID	clrtable_id	Identifier for the colour table		Integer		*[integer number]*
colour table name	clrtable_name	Name or title of the colour table		varchar(40)		"NDVI Colour Table"

colour table URL	clrtable_url	File with colour tables, legends, etc		varchar(80)		"\\nrdb\Colours\NOAANDVI.clr"
colour table format	clrtable_format	The format of the colour table or legend file		varchar(15)	[Windisp / ESRI BIL or AVL]	"WinDisp"
Section 6						
Source table						
Source ID	source_id	An identifier for the source		Integer		*[integer number]*
Source Name	source_name	The name of the source of the data		varchar(40)		" FAO_ARTEMIS"
Source contact ID	source_contact	An identifier for the contact at the source		Integer	integer number representing contact	*[integer number]*
Distribution channel		The channel used to distribute the images		varchar(40)	[e-mail, ftp]	"FTP"
Source URL	source_url	URL for the source, to include username and password	linkage	varchar(80)		"ftp.ext-ftp.fao.org"
Compression type	compression_type	Type of compression or software used to compress the images		varchar(15)	[Null or software used]	"WinZip"
Compression password	compression_passwd	[for compressed files]		varchar(15)		*[password text]*
Comments	source_comments	General comments on the source of the data		Free text		"source allows anonymous login"
Section 7						
Decoder software table	*This table contains information about the main software entrusted with the viewing and handling of an image format*					
decoder ID	decoder_id	ID of the image decoding software		Integer		*[integer number]*
decoder name	decoder_name			varchar(25)		"WinDisp"
Platform	platform	The operating system on which the decoder works		varchar(25)	Windows, Linux, etc	"Windows"

URL of decoder	decoder_url	(network link)		varchar(80)		"\\nrdb\software\ WinDisp.exe"
Command line	commandline	Whether the software can executed in command prompt		Boolean	[Yes/no]	"No"
Usage information	usage_info	Information on how to use the decoder		Text	Free text	[N/A]
Section 8						
Distribution Information table	*The distribution table was basically created to hold information relevant the distribution 'scheme' for each image data type; for example method of distribution (e-mail, FTP), image format used, compression type, the person responsible for distributing the image data.*					
Distribution ID	distribution_id	Identifier for the distribution scheme		Integer		*[integer number]*
Distribution name	distribution_name	Arbitrary name for the distribution scheme		varchar(40)		"GAC_Data_Distribution"
Distribution format	distribution_format	The format in which the images are distributed	distFormat	Integer	From list of available formats	*[integer number]*
Distribution Channel	distribution_channel	The mode via which the images are distributed	medName	varchar(40)	[E-mail, FTP]	"FTP"
Compression type	compression_type	Software used to compress the images		varchar(15)		"WinZip"
Distributor	distributor	Contact ID for person responsible for distribution	distributor	Integer		*[integer number]*
Distribution URL	distribution_url	URL of the network resource used to distribute the images		varchar(80)	URL	"http://www.sadc.int/geonetwork"
Comments	distribution_comments	General comments on the distribution scheme		text	Free text	"data distributed every ten days"
Section 9						
Contacts Table						

contact ID	contact_id	Identifier for the contact		Integer		*[integer number]*
Contact name	contact_name	Name of the contact		varchar(40)		"Blessing Siwela"
Contact e-mail	contact_email	E-mail address of the contact		varchar(40)		"bsiwela@sadc.int"
Contact physical address	physical_address	The full physical address of the contact		text	Free text	" plot 116, millenium park"
Contact postal address	postal_address	The full postal address of the contact		text	Free text	"p. bag 0095, Gabs"
Contact Telephone	contact_telephone	Telephone number(s) for the contact		text		"+6055946344"
Contact website	contact_website	The website of the contact or their organisation		varchar(80)		earlywarning.usgs.gov
Contact role	contact_role	The role of the contact		varchar(50)	[metadata author | datasetContact | distributor | dataSupplier]	"author"
Comments	contact_comments	General comments on the contact		text		"contact works from home on Fridays"
Section 10						
Image format table						
Format ID	formatid	Identifier for the image format		Integer		*[integer number]*
Format name	formatname	The name of the image format	formatName	varchar(40)		'WinDisp IDA'
Format short name	formatshortname	The short name of the image format		varchar(10)	GIF, IDA, TIFF, etc	'IDA'
Format description	formatdescription	A description of the image format		text	Free text	*'Image Display and analysis (IDA) is a binary image format with 512 bytes at the start of the image file containing*

						header or image parameter information.'
Number of bits per pixel	nbitsperpixel	Number of bits used to represent an image value	bitsPerValue	Integer	8, 16, 24, 32	8
Number of Bands	nbands	Number of bands in the image		Integer		1
Decoder ID	image_decoder	The ID of the software that can decode this image type		Integer		*[integer number]*
Section II						
Image table						
image id [pk]	image_id	A unique identifier for the image		Integer		*[integer number]*
image name	image_name	Name of the image		varchar(50)		"V08103"
image title	image_title	Title of the image	title	varchar(80)		"vegetation index for October 21-31, 2008"
Image URL	image_url	Complete URL of the image	dataSetURI	varchar(100)		"\\nrdb\RSData\NDVI\V08103.iml"
File size	image_size	File size in kilobytes		Integer		506KB
image type ID	image_type	Identifier for the image type		Integer		*[integer number]*
Creation date	creation_date	The date the image was created	resRefDate	Date		"2008-11-01"
Start of time period of content	time_period_start	The start of the time period for which the image applies	exTemp	Timestamp		"2008-10-21"
End of time period of content	time_period_end	The end of the time period for which the image applies	exTemp	Timestamp		"2008-10-31"
Date of acquisition	acq_date	The date on which the department acquired the image		Date		"2008-11-02"

Cloud cover measure	metadata_id	Cloud cover percentage estimated or calculated	cloudCovPer	Integer		25%
browse graphic filename	bgraphic_filename	Name of the browse graphic		varchar(40)		"v08103.gif"
browse graphic format	bgraphic_format	The file format of the browse graphic		varchar(5)	BLOB, GIF, JPEG, PNG	"GIF"
browse graphic URL	bgraphic_url	The location of the browse graphic		varchar(80)		\\nrdb\RSdata\NDVI\v08103.gif"
Comments	image_comments	General comments on the quality, bad lines, etc		text	Free text	"image has some missing lines"
metadata schema	metadata_schema	The name of the metadata schema in use		varchar(20)		"ISO19115"
Metadata language	metadata_language	The language used to document the metadata	mdLang	varchar(15)		English
metadata creation date	metadata_date	The date the metadata record is created	mdDateSt	Date		"2008-11-03"
Metadata Contact ID	metadata_contact	ID for metadata author	mdContact	Integer	List of authors from database	*[integer number]*

Appendix E - Metadata Entry Forms

Data Type Entry	
Data type Name	*
Data content type	*
Data purpose	*
Data abstract	
Data Theme	--choose Theme--
Keywords	
Time series	--choose TimeSeries--
Data Lower limit	
Data Upper limit	
Data extent [polygon]	Upper-left Upper-right Lower-right Lower-left
Data status	Completed
Data citation	
Data credit	
Contact	--choose Contact--
Main source	--choose Source--
Alternative source	--choose Source--
Data language	English
Data distribution scheme	--choose Distribution scheme--
Maintenance frequency	
Usage constraints	
Comments	
Submit	Reset

Image / Data source Entry	
Source Name	
Distribution channel	E-mail [e-mail, FTP]
Source URL	
Compression	None
Compression Password	
Source Contact	Ronald Smith [chose contact person from list]
Comments	
Submit	Reset

(Software) Decoder Entry	
Decoder Name	
Decoder platform	Windows
Decoder location [URL]	
Commandline?	Yes
Usage	
Comments	
Submit	Reset

Distribution Entry	
Distribution Name	
Distribution Channel	E-mail
Distribution URL	
Distribution Format	WinDisp IDA
Compression type	None
Distributor / contact	--choose Contact--
Comments	
Submit	Reset

Image Format Entry	
Format Name	
Format short name	
Format description	
Number of bands	
Number of Bits per Pixel	8
Format Decoder	WinDisp
Submit	Reset

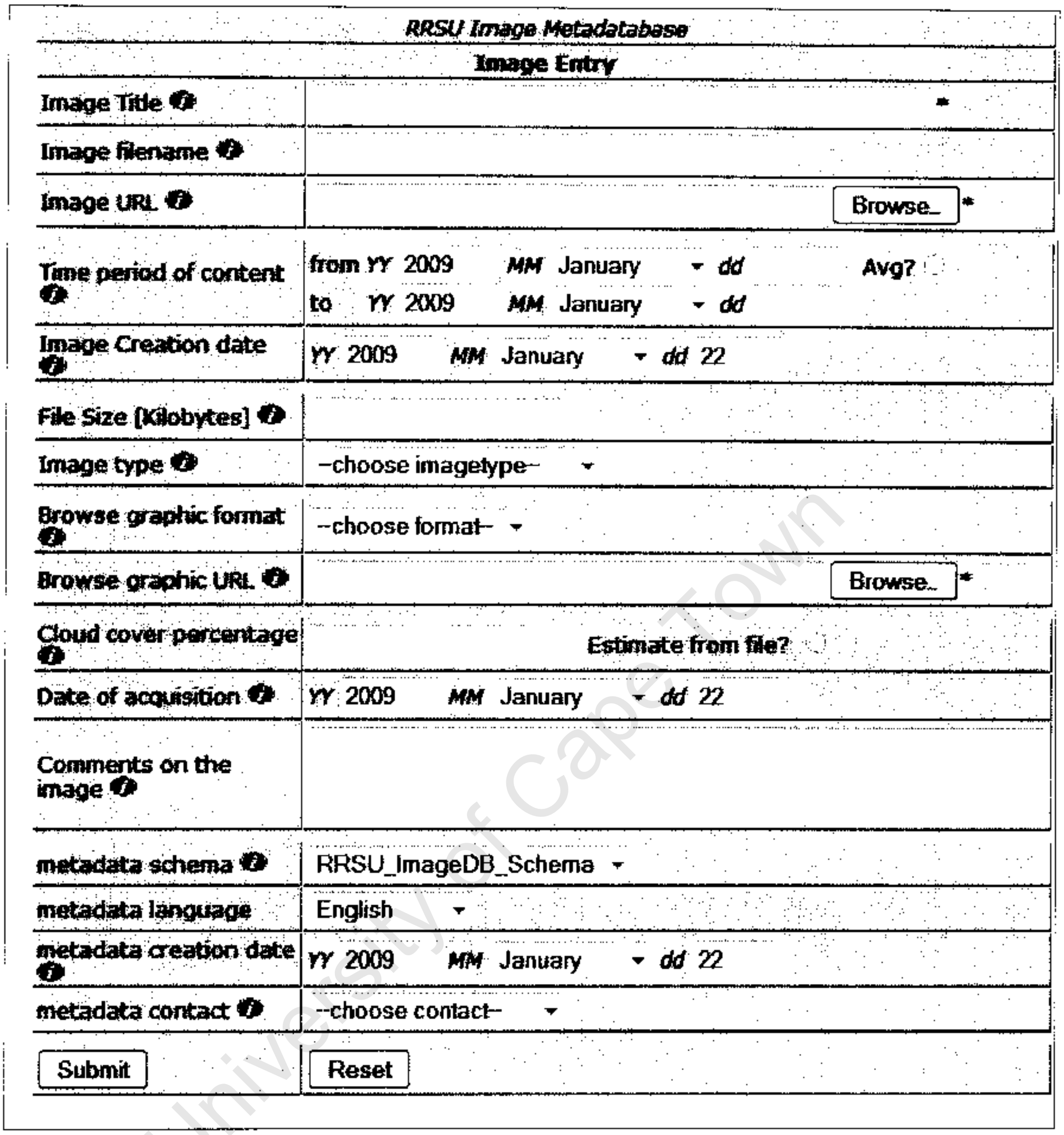
Image Type Entry	
Image type Name	*
Spatial Resolution (X)	*
Spatial Resolution (Y)	*
Image Rows [height]	*
Image Columns [width]	*
Image format	WinDisp IDA
Image slope	
Image intercept	
Image lower limit	
Image upper limit	
Image Missing Value	
Longitude Centre	
Latitude Centre	
Image X Centre	
Image Y Centre	
Map Projection	Geographic
Image Data type	CPC_RFE_SADC
Image Colour Table	NOAA NDVI colour table
Submit	Reset

Theme Entry	
Theme Name	
Comments	
Submit	Reset

Time series Entry	
Time series Name	
Time series Start date	Year: Month: January Day:
Time series End date	Year: Month: January Day: Ongoing ○
Time series interval	10-day
Comments	
Submit	Reset

Colour table Entry	
Colour Table Name	
Distribution URL	
Colour Table Format	Windisp
Submit	Reset

Contact Entry	
Contact Name	
Contact Email	
Postal Address	
Physical Address	
Contact Telephone	
Contact website	
Role of contact	☐ metadata author ☐ dataset contact ☐ dataset distributor ☐ dataset supplier
Comments	
Submit	Reset


RRSU Image Metadatabase
Image Entry
Image Title
Image filename
Image URL
Browse...
Time period of content
from YY 2009 MM January dd
to YY 2009 MM January dd
Avg?
Image Creation date
YY 2009 MM January dd 22
File Size [Kilobytes]
Image type
--choose imagetype--
Browse graphic format
--choose format--
Browse graphic URL
Browse...
Cloud cover percentage
Estimate from file?
Date of acquisition
YY 2009 MM January dd 22
Comments on the image
metadata schema
RRSU_ImageDB_Schema
metadata language
English
metadata creation date
YY 2009 MM January dd 22
metadata contact
--choose contact--
Submit
Reset

Appendix F - SQL statements for creation of database and tables

```
CREATE DATABASE rrsu_imagedb WITH ENCODING='UTF8'
OWNER=postgres;

CREATE TABLE contact (
        contact_id SERIAL PRIMARY KEY,
        contact_name varchar(40) NOT NULL,
        postal_address text,
        physical_address text,
        contact_email varchar(40) NOT NULL,
        contact_telephone text,
        contact_website varchar(80),
        contact_role varchar(50) NOT NULL,
        contact_comments text
)  WITHOUT OIDS;

CREATE TABLE theme
(
        theme_id SERIAL PRIMARY KEY,
        theme_name varchar(40) NOT NULL,
        theme_comments text
) WITHOUT OIDS;

CREATE TABLE colourtable
(
        clrtable_id serial NOT NULL,
        clrtable_name character varying(40) NOT NULL,
        clrtable_url character varying(80) NOT NULL,
        crrtable_format character varying(15) NOT NULL,
        CONSTRAINT colourtable_pkey PRIMARY KEY (clrtable_id)
) WITHOUT OIDS;

CREATE TABLE decoder
(
        decoder_id SERIAL PRIMARY KEY,
        decoder_name varchar(25) NOT NULL,
        platform varchar(15) NOT NULL,
        decoder_url varchar(80),
        commandline boolean,
        usage text
) WITHOUT OIDS;

CREATE TABLE timeseries
(
  tseries_id SERIAL PRIMARY KEY,
```

```
  tseries_name varchar(40) NOT NULL,
  tseries_start date NOT NULL,
  tseries_end date,
  tseries_interval integer NOT NULL,
  tseries_comments text
) WITHOUT OIDS;

CREATE TABLE imageformat
(
  formatid SERIAL PRIMARY KEY,
  formatname varchar(40) NOT NULL,
  formatshortname varchar(10) NOT NULL,
  formatdescription text,
  nbands integer NOT NULL,
  nbitsperpixel integer NOT NULL,
  format_decoder integer NOT NULL references decoder(decoder_id)
) WITHOUT OIDS;

CREATE TABLE projection
(
  projection_id SERIAL PRIMARY KEY,
  projection_name varchar(40) NOT NULL,
  projection_units varchar(10) NOT NULL,
  central_meridian double precision,
  latitude_origin double precision,
  sparallel1 double precision,
  sparallel2 double precision,
  false_easting double precision,
  false_northing double precision,
  scale_factor double precision,
  ellipsoid varchar(20),
  datum varchar(20)
) WITHOUT OIDS;

CREATE TABLE source
(
  source_id SERIAL PRIMARY KEY,
  source_name varchar(40) NOT NULL,
  distr_channel varchar(40) NOT NULL,
  source_url varchar(80),
  compression_type varchar(15),
  compression_passwd varchar(15),
  source_contact integer references contact(contact_id),
  source_comments text
) WITHOUT OIDS;

CREATE TABLE distribution
(
  distribution_id SERIAL PRIMARY KEY,
  distribution_name varchar(40) NOT NULL,
```

```
  distribution_channel varchar(40) NOT NULL,
  distribution_url varchar(80),
  distribution_format integer,
  compression_type varchar(15),
  distributor integer references contact(contact_id),
  distribution_comments text
) WITHOUT OIDS;

CREATE TABLE datatype
(
  datatype_id SERIAL PRIMARY KEY,
  datatype_name varchar(25) NOT NULL,
  data_content varchar(40) NOT NULL,
  data_purpose varchar(40) NOT NULL,
  data_abstract text,
  data_theme integer NOT NULL references theme(theme_id),
  data_keywords varchar(60) NOT NULL,
  data_timeseries integer references timeseries(tseries_id),
  data_lolimit double precision,
  data_uplimit double precision,
  data_status varchar(15),
  data_citation text,
  data_credit text,
  data_contact integer NOT NULL references contact(contact_id),
  data_source integer references source(source_id),
  data_altsource integer references source(source_id),
  data_language varchar(10),
  data_distribution integer references distribution(distribution_id),
  data_maintFreq varchar(30),
  data_usageCons text,
  data_comments text
) WITHOUT OIDS;

//the following SQL statement adds a geometry column 'data_extent' to the table 'datatype'
SELECT AddGeometryColumn('datatype', 'data_extent', 5001, 'POLYGON',2);

CREATE TABLE imagetype
(
  imagetype_id SERIAL PRIMARY KEY,
  imagetype_name varchar(25) NOT NULL,
  spatialres_x double precision NOT NULL,
  spatialres_y double precision NOT NULL,
  image_rows integer NOT NULL,
  image_columns integer NOT NULL,
  image_format integer NOT NULL,
  image_slope double precision NOT NULL,
  image_intercept double precision NOT NULL,
  image_lolimit integer,
  image_uplimit integer,
```

```
  image_missingv integer,
  long_centre double precision,
  lat_centre double precision,
  image_xcentre double precision,
  image_ycentre double precision,
  image_projection integer NOT NULL references spatial_ref_sys(srid),
  image_clrtable integer references colourtable(clrtable_id),
  image_datatype integer NOT NULL references datatype(datatype_id)
) WITHOUT OIDS;

CREATE TABLE image
(
  image_id serial NOT NULL,
  image_name character varying(50) NOT NULL,
  image_title character varying(80) NOT NULL,
  image_url character varying(100) NOT NULL,
  image_type integer references imagetype(imagetype_id),
  image_size integer,
  bgraphic_filename character varying(40),
  bgraphic_format character varying(5),
  bgraphic_url character varying(80),
  time_period_start date,
  time_period_end date,
  creation_date date,
  acq_date date,
  cloud_cover integer,
  image_comments text,
  metadata_schema character varying(20),
  metadata_language character varying(15) NOT NULL,
  metadata_date date NOT NULL,
  metadata_contact integer references contact(contact_id),
  CONSTRAINT image_pkey PRIMARY KEY (image_id)
) WITHOUT OIDS;

ALTER TABLE colourtable OWNER TO postgis;
ALTER TABLE contact OWNER TO postgis;
ALTER TABLE datatype OWNER TO postgis;
ALTER TABLE decoder OWNER TO postgis;
ALTER TABLE distribution OWNER TO postgis;
ALTER TABLE image OWNER TO postgis;
ALTER TABLE imagetype OWNER TO postgis;
ALTER TABLE imageformat OWNER TO postgis;
ALTER TABLE projection OWNER TO postgis;
ALTER TABLE source OWNER TO postgis;
ALTER TABLE theme OWNER TO postgis;
ALTER TABLE timeseries OWNER TO postgis;
```

Glossary

DBMS — Database management system — software that controls the organization, storage, retrieval, security and integrity of data in a database. [http://encyclopedia2.thefreedictionary.com/DBMS]

Raster — a data model in which the locations of geographic objects or conditions are defined by the column and row position of the cells that they occupy in a two-dimension grid (Lillesand and Keifer, 2000).

Image — a pictorial representation of photographic or other data, usually presented in raster format. In this document, the terms image and raster data are used interchangeably.

WinDisp - a desktop public domain software package for the display and analysis of satellite images, maps and associated databases, with an emphasis on early warning for food security, and was originally developed for the FAO's Global Information and Early Warning System (Charlier, 1999).

WMS — web map service. A Web Map Service (WMS) produces maps of georeferenced data, where a "map" is defined as a visual representation of the geographic data (OGC, 2001).

Remote Sensing — the acquisition of information or measurement of some property of an object by a recording device that is not in physical or intimate contact with the objects under study (http://www.cdmha.org/definitions.htm).

RRSU — Regional Remote Sensing Unit. This unit collects data from remote sensing systems and uses this data in applications related to food security and natural resources management.

FAO — Food and Agriculture Organization of the United Nations

GIEWS — Global Information and Early Warning System of the UN's FAO.

GIEWS Workstation — database and software developed by GIEWS for the handling of information used in early warning for food security.

GeoNetwork — GeoNetwork Metadata Catalogue. A metadata management system developed by FAO for geographic datasets which uses the ISO 19115 metadata standard.

JDBC — Java Database Connectivity