FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION

ITC

WEB PROCESSING SERVICES FOR A FARM SIMULATION MODEL

HIWOT BERHANE
March, 2011

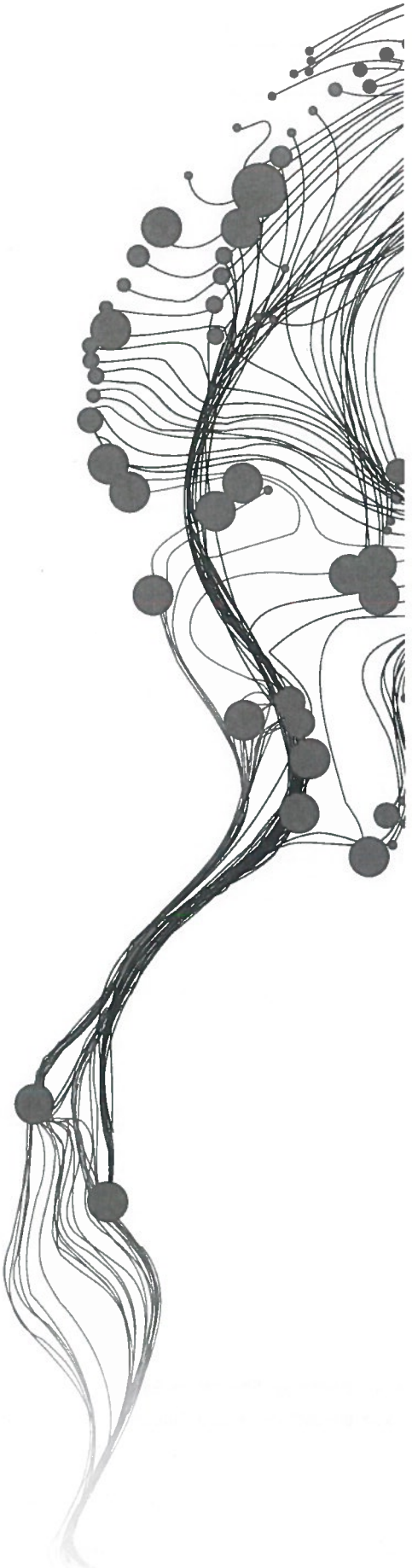
SUPERVISORS:

Dr. Ir. R.A. (Rolf) de By
Dr. R. Zurita-Milla

UNIVERSITY OF TWENTE.

ITC

RC 33899/33800



WEB PROCESSING SERVICES FOR A FARM SIMULATION MODEL

ITC Library
www.itc.nl/library • P.O.Box 6 • 7500 AA Enschede • The Netherlands
ITC

HIWOT BERHANE
Enschede, The Netherlands, March, 2011

Thesis submitted to the Faculty of Geo-information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: Geoinformatics

SUPERVISORS:

Dr. Ir. R.A. (Rolf) de By
Dr. R. Zurita-Milla

THESIS ASSESSMENT BOARD:

Dr. Ir. R.A. (Rolf) de By(chair)
Dr. Ir. M. van Keulen

ABSTRACT

Agriculture globally faces a continuously changing environment due to trade, globalization, climate change and social needs. It is becoming difficult for farmers to choose which type and method of farming to use due to this changing world. Agricultural simulation models try to simulate this complex reality to help farmers in their decision making, by providing them estimation on crop production. Most simulation models are standalone applications that require an expert user who is aware of its diverse input parameters, GIS operations (functionality) and IT technology. In this study we propose to design and implement a web processing service for our farm simulation model (FSSIM) in an open service-oriented architecture (SOA).

All of the diverse model input parameters could be accessed from different catalogue servers through metadata search. However, the resulting datasets from the search cannot be used directly, because the data may be in different format, projection, scale, thematic content, perhaps even-up-to-datedness; such data may need to be transformed first. We focused our research on structural heterogeneity within the collected datasets and design a conceptual schema mapping theory for it. The conceptual schema mapping theory tried to harmonize the heterogeneous datasets to the unified or target schema that the model uses. In addition, a comparative study has been made between our theory and already existing open source schema mapping software, HALE (graphical user interface that help to formulate a mapping rule) and CST-WPS which makes the actual data transformation.

We have shown in this research using our case study that the conceptual theory that we made and the actual schema mapping implementation from HALE have a slightly different semantics. This means all of our theories on the conceptual schema mapping operations are implemented in HALE software. However, some of them operate on different levels of the schema than we designed.

Keywords

Schema mapping on web service, Heterogeneous datasets on web service

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
Introduction

1.1 MOTIVATION AND PROBLEM STATEMENT

Agriculture globally faces a continuously changing environment due to trade, globalization, climate change and social needs. It is becoming difficult for farmers to choose which type and method of farming to use due to this changing world. Research groups are developing simulation models to help farmers in their decision making, by providing them estimation on crop production. These models take a number of inputs, which include environmental features, economic parameters, socio-cultural values and management issues [19]. Most farm simulation models are standalone applications that require a literate person who is aware of its diverse input parameters, GIS operations (functionality) and IT technology. However, there should be special attention when developing a farm simulation model for Sub-Saharan countries because most farmers in that region are less educated when compared the developed countries in the world. We can hardly say that the existing simulation models are useful for Sub-Saharan African farmer communities since they are unable to understand and use the model efficiently.

A farm simulation model is being built for Burkina Faso (FSSIM) at ITC for optimizing crop productivity plans. The model requires diverse input parameters for making the analysis, which will be acquired from different organizations. These data providers for example are, FAO, which offers soil datasets, NASA power agro-climate, which provides weather information, Agristat, which provides sub-national statistics at the province (district) level and SPAM, which provide production information. These datasets can be discovered through metadata search on catalogue services. The resulting datasets from the search cannot be used directly, because the data may be in different format, projection, scale, thematic content, perhaps even-up-to-datedness; such data may need to be transformed first. The model also needs expert or skilled users for locating, gathering and merging all the appropriate geospatial and non-spatial datasets. There should be a mechanism to present FSSIM in such a way that the unskilled farmer in Burkina Faso would find easy to use.

The farm simulation model (FSSIM) should shift from standalone or desktop application towards distributed geospatial web services, which are based on service-oriented architecture (SOA). In addition, "the services should be designed to be well-defined, self-contained and self-descriptive so that they can be found and accessed via available discovery mechanisms" [14]. There should be a web processing service (WPS) to execute the model. "Since a WPS interface standardizes the way processes and their inputs/outputs are described and standardize how a client can request the execution of a process and how the output from a process is handled" [34]. This is to allow the simple users to make use of the technology and FISSIM without being aware of the complexity of the IT environment or the diverse input parameters that the model requires.

A Schema mapping service is one of the services that should be integrated in the WPS to create harmonized data integration between the different datasets. The integration is made possible by mapping a source dataset with different schema to a different target schema [28]. It is necessary to have standardized and generic mechanism that maps the different datasets to the format required by the FSSIM model.

1.2.2 Research questions

To achieve the objectives, the following research questions need to be answered:

1. How to translate simple input from an user to fetch spatial data from different catalogues?
2. What is the best mechanism for schema mapping of different dataset to a schema that the FSSIM model requires?
3. What is the best way to initialize, connect and transfer message between services on executing service chaining of the WPS workflow?

1.2.3 Innovation aimed at

This research has the aim of optimizing the efficiency and usability of FSSIM model by using service chaining to integrate the different web processing services. We will implement a schema transformation technique to combine the heterogeneous dataset requirement of FSSIM model and make it interoperable with the WPS workflow.

1.2.4 Related work

Several researches have been done to resolve the problem of heterogeneous datasets through schema mapping. A manual schema mapping is one of the methods that have been applied to solve this problem [21]. "Manually specifying schema mapping is a tedious, time-consuming, error-prone, and therefore expensive process and the level of effort is linear in to the number of matches to be performed " [38]. In addition, manual schema matching also requires operators to have enough knowledge on the different datasets to be mapped. This work deviated from our research that it needs an expert user to make the schema mapping.

A project in UK called VISITA tries to harmonize the different datasets by using semi-automated a schema mapping tool, FME (Feature Manipulation Engine) for their research [7]. The project integrates different datasets to a common or global schema and provides the result as web feature service but still needs a person who provide the source schema and the target schema for the schema mapping process. Our alternative implementation differs from this project by providing total automatic implementation of the schema transformation in a web service so that it does not require any involvement of end user on the schema mapping process.

A research project has been done between ITC and Finnish geodetic institute that tries to use schema mapping tool and generalization technique in a web service environment. The project uses the schema mapping tool to rename attributes and transform between coordinate and also uses a generalization technique to collapse polygon geometry to point. The project uses service chaining to connect the schema mapping service, generalization service and WMS (web map service) to provide the final map [15]. This research only considers one schema transformation operation, renaming operation. This point differentiate it from our project.

1.3 METHOD ADOPTED

To meet the research objectives, we will follow the following methodology:

- Literature review on web processing services, schema mapping and service chaining.
- Collect necessary datasets for FISSIM model from different organization (data providers).

Chapter 2
Literature Review

2.1 HETEROGENEOUS DATASETS

The objective of this research is to search for appropriate datasets that the FSSIM model requires but datasets exist in different format, interface and structure in different SDI nodes than the model needs. The heterogeneity of the datasets makes it very difficult for us and also for whole geographical society to use the available datasets. According to [8] the existence of heterogeneous datasets is due to :

1. Syntactic heterogeneity:occurs when the objects the different data format [3]. The same dataset can be stored in different storage format, for example as, relational database, shape file or as GML document [3].

2. Semantic heterogeneity: occurs when the same object, has more than one name in different organization. There are two types of semantic heterogeneities, naming and cognitive heterogeneity [12]. Naming mismatch refers to when the same object has different names in a different organization. It can be corrected through simple mapping, using a thesaurus [12]. Cognitive heterogeneity means the same object can have multiple role in different organization [12].

3. Schematic or data model heterogeneity: Schema is a description of a dataset; it is not the actual data but a description or structure of spatial and non-spatial datasets. Schematic heterogeneity occurs due to the structure, hierarchy and classification of a real object entity varies among organizations or disciplines. According to [7], schematic heterogeneity or conflict arises due to;

    - Structural semantics: elements at the schema level may have the following mismatches.
        - Type mismatch, occurs when the same class or object are assigned different data type in different organization or disciplines.
        - Range mismatch, occurs when an organizations use different data value ranges fro the same object. For example one organization may use an integer range from 0-100 and another may use for the same object a data value from -100 to 100.
        - Some attributes would only exist in the target schema but no correspondence (missing attribute) in the source schema.
        - Classification mismatch, spatial and non-spatial attribute classification of source schema is different from target schema. For example source schema may classify land use as in settlement; forest and water area but target schema may have more number of land use classes.
    - Granularity mismatch, occurs when different organization uses different level of detail to represent their entity. For example an organization may represent a main road

2. Hypertext Transfer Protocol (HTTP) allows communication between web browsers, servers and related web applications. HTTP has two types of messages to communicate between computers (client and server component). These are request and response which both have similar structure. The request is a message send from client computer to sever asking for resource or service and the response is a message sent from server to client answering the specified request [39].

3. Web Service Description Language (WSDL) is an XML document used to describe the location of a service and the operation that the service performs. WSDL makes the services to be self-describing, easier to be invoked and easier to be used by other services.

4. Simple Object Access Protocol (SOAP) is XML based communication protocol for exchanging information between applications.

5. Universal Description, Discovery and Integration (UDDI) provide an interface to describe and identify web services and web services providers [11]. It stores a description of services in the form of WSDL description. UDDI register the web service metadata including or the WSDL description of services. And this allows the client to search for the service they want and to access it.

2.3 SOA AND WEB SERVICES

SOA provides an architecture where web services communicate between each other. The communication could be to obtain data service or processing services. Some web service provides access to certain processing operation through a standardize interface and some offers datasets on a particular theme. SOA architecture composed of three participants who are service providers, service requester and service registry and three fundamental operations which are publish, find and bind that facilitate the interoperable communications between the services. The service provider makes their service available on the Internet by publishing it on service registry. Service consumer or requester utilizes the existing services by finding and binding the existing services provided by the service provider on the service registry [9].

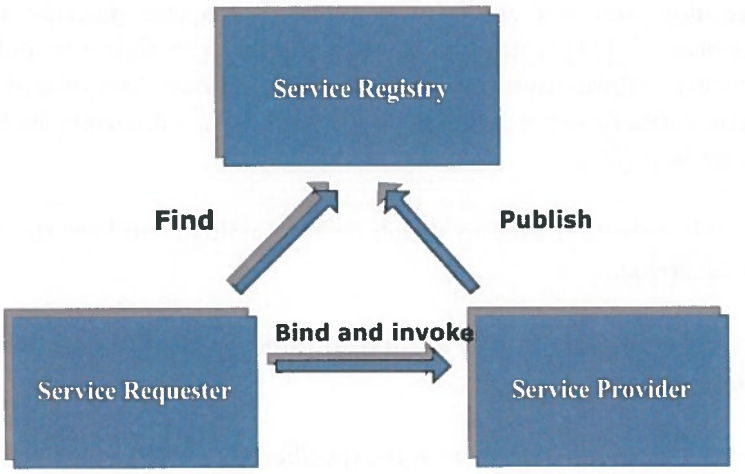


Figure 2.1: Basic operation of SOA (adapted from [31])

According to [4] services can be grouped into three basic categories :

2.3.3 Web Processing Services (WPS)

WPS provide standardize interface to perform any GIS functionalities ranging from simple feature selection to complex calculations."The WPS interface standardizes the way processes and their inputs/outputs are described, how a client can request the execution of a process, and how the output from a process is handled " [34]. There are operations or functions that enable clients to access and use WPS provided by service providers. These operations are described in OGC specification as follows [33],

- GetCapabilities : this operation allows the clients to investigate service by looking at the returned metadata information about the service. The metadata information contains the names and functionalities of each process offered in the WPS.
- DescribeProcess: this operation returns to the requester information about service operation, like input required or allowed and information about the output of the process.
- Execute: this operation returns the result of the process or WPS function that the client requested by providing input parameters.

The processing services provide processing operations based on the user defined parameters. Data harmonization services that we focus on this research used to combine heterogeneous datasets with different coordinate projection, different language and format into consistent, uniform target datasets. The data harmonization services are provided as a form of web processing services. The services are coordinate transformation service and schema mapping service. These services should be synchronized one after the other in order to reach the intended and correct result. WPS profile helps to search for and invoke web processing services that are registered with the service registry and also to make our WPS accessible to other clients or service requesters.

WPS profile

Based on WPS specification, a WPS to be reusable and interoperable it must have standard WPS profiles that contain the following elements [18],

- A Universal Resource Name (URN): is the address of a WPS that uniquely identifying a WPS.
- A Reference response: is a mandatory element that is used for reference a DescribeProcess operation or request [30]..
- A Description: is an optional element of WPS profile which is human readable document used to describe the WPS and its implementation [18].
- A Web Service Description Language (WSDL): is an optional element of WPS profile which is used to describe the process [30]..

2.3.4 Web service chaining

Complex GIS task may not be manipulated by a single web service and it might require many services to orchestrate together to execute the task. one of the benefit of web service is the ideology of web service chinning, that is also the basic framework of service oriented architecture (SOA).

Data integration or harmonization cannot be executed by a single web processing service since it needs different tasks, like generalization, coordinate projection, attribute renaming, and feature selection and so on. This independent web processing services should be chained or orchestrated

Chapter 3

Automating conceptual schema mapping

3.1 CONCEPTUAL SCHEMA TRANSFORMER

Web-based data harmonization process can be looked at in two ways, schema transformation and generalization. Schema transformation harmonizes the thematic characteristics of heterogeneous datasets, whereas generalization transforms the spatial properties and level of detail of datasets [15]. In this chapter We will discuss the Conceptual Schema Transformer (CST), which performs schema transformation to a dataset, as expressed in one specific schema, to a dataset, expressed in another target schema. There are many desktop schema mapping software packages that require expert user involvement to produce the intended mapping rules, which yield a target dataset. CST operations need to be automated in a web service environment to allow data harmonization on the fly without user involvement. A Conceptual Schema Transformer has three inputs: *source schema, source dataset* and *target schema.*

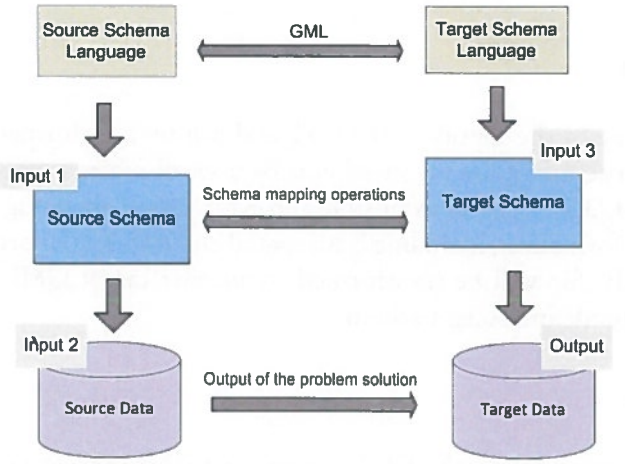


Figure 3.1: Conceptual Schema Transformation

To create an automatic CST, there should be additional information other than the three input documents, because source and target schema do not contain information like scale, coordinate reference system and thematic content of the dataset. This information should be searched for in the metadata that describe the dataset. There should also be a mechanism to maintain the knowledge of expert users that allows smooth schema conversion without them.

3.2 SOURCE AND TARGET SCHEMA

The source and target schema are the two inputs to the CST. The source and target schema contain a set of information; let us consider the source and the target schema as two different data types.

Reclassification The process of reclassifying or classifying a source schema attribute value to a different class based on the number of class requirement of the target schema attribute.

Renaming of feature attributes The process of changing source attribute name into a target attribute name.

Merge of features or attributes The process of concatenating two or more source attribute values to a single target attribute value.

Type conversions The process of changing a source data type to a target data type, and copying source values to respective target values.

Augmentation The process of filling a default or null value to a target schema attribute that has no matching attribute value in the source schema.

3.7 ANALYSIS OF CST OPERATORS

(The section on formal language basis was provided by my supervisor) The source dataset SD has as its type the source schema SS, and thus it contains tables:

$$SD : \langle T_1 : \mathbb{P}\sigma_1, \ldots, T_m : \mathbb{P}\sigma_m \rangle .$$

The target dataset TD, likewise, has as its type the target schema TS, and will eventually also contain tables:

$$TD : \langle T_1 : \mathbb{P}\tau_1, ..., T_n : \mathbb{P}\tau_n \rangle .$$

In the sections below, we define and discuss a small language of powerful expressions that we specifically developed to define schema transformations. Our intuition on that language is based on three important design principles:

- Schema transformations are functions that construct a dataset from a given dataset. Such functions may affect the structure but also the data content.
- We want to achieve a *modularized* transformation language, in which separate transformations can be combined to define an overall transformation; technically speaking, this means that we want to achieve functional *compositionality* of transformations. Simply put, if f_1 and f_2 are two independent transformations of a source dataset SD, then the following equation should hold most of the time:

  $$f_1 \circ f_2(SD) = f_2 \circ f_1(SD),$$

  where $\circ$ denotes functional composition. We are writing "most of the time" because two transformations may not always be as independent from each other as we would wish. We return to discuss this matter in Section 3.8.
- The formal semantics of the expression forms that we propose is defined in such a way that it allows us to express the intuition "take the whole dataset, make the indicated local change, and return the dataset thus obtained."

Basic expressions needed

This section lists a number of expressions that are needed to define schema transformation expressions. They provide the fundamental 'building blocks' to construct larger, end-user-oriented expressions. Assume $r : \langle a_1 : \tau_1, \ldots a_n : \tau_n \rangle$ is a record, assume that R_1 and R_2 are tables of type $\mathbb{P}\langle a_1 : \tau_1, \ldots a_n : \tau_n \rangle$ with primary key $\{a_{i_1}, \ldots, a_{i_h}\}$, and that S, T and U are union-compatible value sets. We will use the notation keys(R_i) to denote the set of primary key values present in table R_i.

1. The expression

$$r \textbf{ except } a_j = e_j, \ldots, a_k = e_k \quad \text{where } \{a_j, \ldots, a_k\} \subseteq dom[[r]] \tag{3.1}$$

denotes a record r' obtained by copying r and replacing its attribute value for a_j by the value of e_j, and so on, until the attribute value for a_k, which value is replaced by the value of e_k. The except expression allows local attribute value overwriting. The formal semantics of this expression is a function shaped like $[[r]]$ that maps onto different semantic values for the listed attributes.

2. The expression

$$r \textbf{ dropatt } a_j, \ldots, a_k \quad \text{where } \{a_j, \ldots, a_k\} \subseteq dom[[r]] \tag{3.2}$$

denotes a record r' obtained by copying r and removing from it the listed attributes. The formal semantics of this expression is a function shaped like $[[r]]$ but with a reduced domain, mapping onto the same values for remaining attributes.

3. The expression

$$r \textbf{ addatt } a_j = e_j, \ldots, a_k = e_k \quad \text{where } \{a_j, \ldots, a_k\} \cap dom[[r]] = \emptyset \tag{3.3}$$

denotes a record r' obtained by copying r and adding to the record the named attributes with associated values. The formal semantics extends that or r by adding to the domain of the function, and mapping onto the values of the respective expressions.

4. The expressions

$$S \textbf{ union } T, S \textbf{ setminus } T, S \textbf{ intersection } T \tag{3.4}$$

denote the regular and well-known set-theoretic expressions.

5. The expression

$$R_1 \textbf{ exceptset } R_2 \quad \text{where keys}(R_2) \subseteq \text{keys}(R_1) \tag{3.5}$$

denotes a set R' obtained by copying R_1 and replacing those records that have a key in keys(R_2) by the corresponding records in R_2. Observe the similarity with the except expression, and the fundamentally different semantics.

$$SD \textbf{ renametable } a \textbf{ to } b. \tag{3.10}$$

The meaning of this expression can be given as a combination of our base expressions:

$$(SD \textbf{ addatt } b = SD.a) \textbf{ dropatt } a.$$

Renaming attributes of a table is also an important schema transformation operation. It allows us to rename an attribute of a source table. Let us assume that T is a table name in SD, that a is one of its attributes, but that b is not. The general form is then

$$SD \textbf{ renameatt } T \textbf{ from } a \textbf{ to } b. \tag{3.11}$$

The meaning of this expression can be given as a combination of our base expressions, through the following equivalent expression:

$$SD \textbf{ except } T = \{\, t \in SD.T \bullet (t \textbf{ addatt } b = t.a) \textbf{ dropatt } a \,\}.$$

3.7.2 Adding and dropping attributes

Adding tables as attributes in our dataset, let us assume that $a_j, ..., a_k$ are table names and are not in SD. The general form is then,

$$SD \textbf{ addatt } a_j = e_j, \ldots, a_k = e_k$$

Dropping tables as attributes from our dataset, let us assume that $a_j, ..., a_k$ are table names in SD. The general form is then,

$$SD \textbf{ dropatt } a_j, \ldots, a_k.$$

Adding table attributes in one of the tables in our dataset, let us assume that T is a table name in SD and $a_j, \ldots, a_k$ are not attributes of T. The general form is then

$$SD \textbf{ addatt } a_j = e_j(t), \ldots, a_k = e_k(t) \textbf{ on } T \textbf{ as } t \tag{3.12}$$

The meaning of this expression can be given as a combination of our base expressions, through the following equivalent expression:

$$SD \textbf{ except } T = \{\, t \in SD.T \bullet (t \textbf{ addatt } a_j = e_j(t), \ldots, a_k = e_k(t))\}.$$

The $e_j(t), \ldots, e_k(t)$ are tuple variables to have a different value for each tuple in the table T.

Dropping attributes in one of the tables in our dataset, let us assume that T is a table name in our dataset SD and $a_j, \ldots, a_k$ are attributes of table T. The general form is then

$$SD \textbf{ dropatts } a_j, ..., a_k \textbf{ from } T \tag{3.13}$$

3.7.5 Type conversion operator

The type conversion operator converts the data type of source dataset attribute to a data type that is specified on the target schema. The conversion could be a spatial type conversion or non-spatial type conversion. Let us assume that T is a table name in SD and a is an attribute of table T,

$$SD \textbf{ converttype } a \textbf{ to } \gamma \textbf{ in } T \tag{3.17}$$

The meaning of this expression can be given as a combination of our base expressions, through the following equivalent expression:

$$((SD \textbf{ addatt } a' = conv(a, \gamma) \textbf{ on } T \textbf{ as } t) \textbf{ dropatt } a \textbf{ from } T) \\ \textbf{renameatt } T \textbf{ from } a' \textbf{ to } a.$$

The function $conv(a, \gamma)$ takes expression a and changes the data type of a by γ, where γ is the data type of the target dataset required by the target schema.

3.7.6 Reclassification operator

The reclassification operator maps values of a source attribute value to a fixed target attribute value. The target dataset may need a certain number of classes and the target schema should contain information on the allowable classification values;

$$SD \textbf{ reclassify } a = f(a) \textbf{ in } T \tag{3.18}$$

The meaning of this expression can be given as a combination of our base expressions, through the following equivalent expression:

$$SD \textbf{ except } T = \{\, t \in SD.T \bullet (t \textbf{ except } t.a = f(a)) \,\}.$$

The function $f(a)$ takes the attribute a and returns the classified value of a.

3.7.7 Augmentation operator

The augmentation operator is used to define values for attributes in the target dataset properties missing in source dataset. The operator assign default value of the attribute in the target dataset since the attribute was not available in the source dataset. The default values of the attributes should be defined in the target schema. This operator is the same as adding a table attributes in equation 3.12.

3.8 COMPOSITIONALITY

As we have seen in Section 3.7, functional compositionality may not exist between the CST operations. We are interested in $f \circ g = g \circ f$, this generally gives us a "yes" or "no" answer but more often it is a no answer and we want to know specifically under what condition it is a yes answer. In the matrix below we will see the condition where functional compositionality exist and where it does not.

The second is where f is applied to the data set before g:

$$g \circ f = (SD \textbf{ mergeatt } a_j, \ldots, a_k \textbf{ to } b = e(a_j, \ldots, a_k) \textbf{ in } T) \textbf{ reclassify } a = f(a) \textbf{ in } T.$$

Also means
g$\circ f$ = (SD **mergeatt** $a_j, \ldots, a_k$ **to** $b = e(a_j, \ldots, a_k)$ **in** T where
$\{a_j, \ldots, a_k\} \in dom[[t]]$ **on** T **as** t)
reclassify $a = f(a)$ **in** T **as** t **where** $a \in dom[[t]]$)

We can say that these two expressions are equivalent, *provided* that the merging function $e(a_j, \ldots, a_k)$ does *not* use the attribute a. If it does, then the reclassification in the second expression does not make sense.

- Case 3
Let us see two schema transformations f and g, and $f \equiv 3.17$ and $g \equiv 3.19$ where $f \equiv SD$ **converttype** a **to** γ **in** T and $g \equiv SD$ **addatt** $a_j = e_j(t), \ldots, a_k = e_k(t)$ **on** T **as** t. To study the compositonality, we need to look at two expressions. The first is

$$f \circ g = (SD \textbf{ addatt } a_j = e_j(t), \ldots, a_k = e_k(t) \textbf{ on } T \textbf{ as } t) \textbf{ converttype } a \textbf{ to } \gamma \textbf{ in } T.$$

This means

fo$g$ = (SD **addatt** $a_j = e_j(t), \ldots, a_k = e_k(t)$ where $\{a_j, \ldots, a_k\} \cap dom[[t]] = \emptyset$ **on** T **as** t)
((**addatt** a' = conv (a,γ) **on** T **as** t) **dropatt** a **from** T)
renameatt T **from** a' **to** a **where** $a \cap dom[[t]] = \emptyset$)

The second is where f is applied to the data set before g:

$$f \circ g = (SD \textbf{ converttype } a \textbf{ to } \gamma \textbf{ in } T) \textbf{ addatt } a_j = e_j(t), \ldots, a_k = e_k(t) \textbf{ on } T \textbf{ as } t.$$

This also means

g$\circ f$ = ((SD **addatt** $a' = conv(a, \gamma)$ **on** T **as** t) **dropatt** a **from** $Twherea \subseteq dom[[t]]$)
renameatt T **from** a' **to** a **where** $a \cap dom[[t]] = \emptyset$))
((**addatt** $a_j = e_j(t), \ldots, a_k = e_k(t)$ where $\{a_j, \ldots, a_k\} \cap dom[[t]] = \emptyset$ **on** T **as** t))

The above two expressions are equivalent, *provided* that the the two equation works on different attribute but even if they work on the same attribute the output is the same due to the *where* clause that restrict the equations to work on already existing attribute for *convettype* and *addatt* make sure the attribute names does not exist.

Chapter 4

Application scenario for schema transformation on web service

4.1 CASE STUDY

Different environmental models require different types of datasets from different data providers for their environmental assessment. Due to the provider organizational structure, task and policy different data providers offer their datasets in different schema and format. It is difficult for the environmental model to use the heterogeneous datasets that are created from different data providers. A farm simulation model is being built for Burkina Faso (FSSIM), which needs diverse input parameters to optimize crop productivity plans. One of the mechanisms to make data harmonization is schema transformation, and this chapter describes schema transformation in a web service.

4.2 TARGET SCHEMA SELECTION

As we have seen in Chapter Three, schema transformation services require the definition of source and target schemas in order to apply mapping to datasets that is organized within these schemas. The target schema should be designed considering the requirement of the system, standards in order to be compatible with other systems and also the mapping rule to be used in the schema.

4.2.1 Methodology for choosing language for target schema

According to INSPIRE Transformation Network Service, schema description language must be selected on the basis of the following criteria [6];

- Expressiveness: The language should be expressive enough to represent all the concepts and intuition of the schema. It should be able to describe the core schema aspects like
  - Object types, simple properties, cardinality, custom data type, inheritance, aggregation, composition, property constraints, default values, annotations,grouping and application information [6].
- Mapping Compatibility: the schema description language should be compatible with the schema mapping language or schema transformation language going to be used.
- Web Compatibility: the language must be compatible and suitable enough to be used in a web service.
- Technology Independence: "the language should not be tied to any specific vendor or tightly coupled to a specific data-encoding format - it should give a conceptual description of the schema"[6].

4.2.3 Schema mapping language

Transforming a source schema into a target schema requires a set of rules that can be used to implement the actual data transformation. These rules are in the form of;

1. Extensible Stylesheet Language Transformation (XSLT):
   XSLT is well-defined XML based language used for transforming XML documents into any other format. XSLT needs the following documents to make the transformation:

   - Well-formed XML document.
   - XSL document: that contains a style sheet to express how structured content should be presented and transformation template.
   - XSLT parser to perform the transformation.

   According to [23] implementation of a three-page matching table or the above three documents for schema translation in the form of a set of XSLT operations can easily yield a script with thousands of lines. These scripts are hard to create, and also harder to maintain. Therefore another transformation language to express the schema translation rules is required.

2. Ontology mapping language (OML)
   OML is an RDF-based language that allows specifying correspondences between two ontologies. OML provide the possibility to represent complex correspondences independently from the language the ontologies are modeled in. This means the mapping language gives a way to represent all kinds of schema mappings [32]. It can easily be used with different conceptual schema languages, such as GML application Schema, UML or database Schema. OML is also expressive enough to be used for query rewriting and instance transformation.

3. Rule Interchange Format (RIF)
   RIF is a W3C recommendation, highly suited to the task of representation of schema mapping definitions as collections of business rules. RIF provides a good, platform independent language for the interchange of mapping definitions. However, it uses a logical and mathematical expressions rather than programming terminology, with simple IF - THEN construct rules with well-formed Formulas [6].

4.3 HUMBOLDT SOFTWARE

Datasets exist in different format, interface and schema in different SDI nodes. The heterogeneity of datasets makes it difficult for the geographical society to use the available datasets from different data providers. Therefore data harmonization tools are a vital component for efficient and meaningful data integration [22]. Humboldt is one of the tools that can be used to solve this problem. It is open source software that helps for data harmonization. It provides the capability for data integration by offering different services like schema transforming service, spatial coordinate projection service and edge matching service [35].

4.3.1 HALE

The HUMBOLDT Alignment Editor (HALE), is a rich graphical user interface for conceptual schema mapping between two schemas. HALE takes source and target schema and produces mapping rules. The mapping rules are expressed in Ontology Mapping Language (OML), a high level language that allows specifying correspondences between two ontologies. The source and

rule since it can be used in different schema description languages and also it works well with CST.

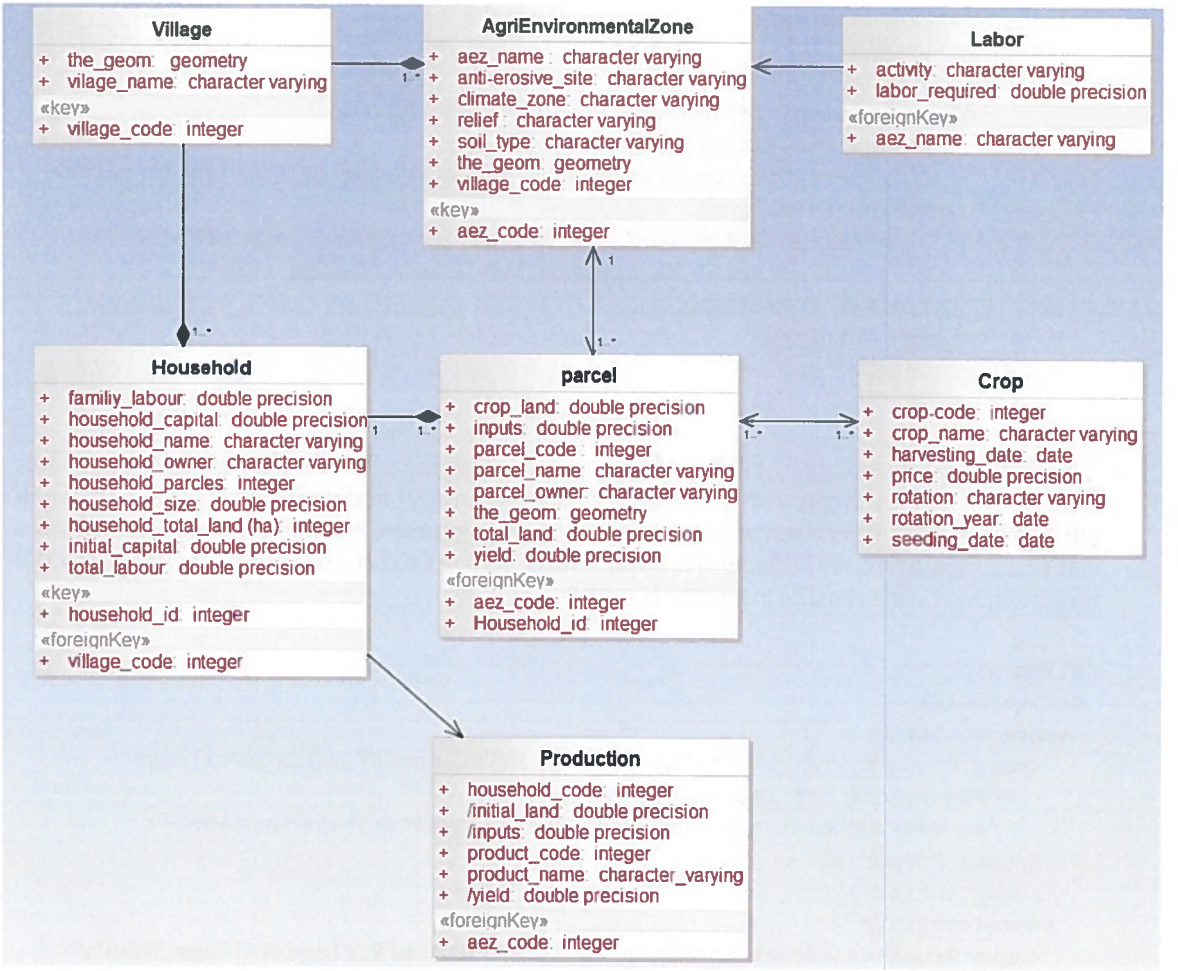


Figure 4.2: Target Schema for FSSIM

4.5 MAKING MAPPING RULE WITH HUMBOLDT FOR FSSIM

Schema mapping from a source schema to a target schema using HALE produces an OML file. The OML file has,

- A header contains onto1 and onto2, which are the addresses of the source and target schema respectively. The header contains a formalism tag, which provides information about the language used to represent the ontologies.

```
<align:level></align:level>
<align:onto1>
    <align:Ontology rdf:about="http://www.MYURL.com">
        <align:location>file:
        /C:/Users/User/Desktop/source/SourceFile2_HST.xml</align:location>
        <align:formalism>
```

```
  <omwg:entity2>
    <omwg:Class rdf:about="http://www.MYURL.com/Village"/>
  </omwg:entity2>
</align:Cell>
```

The above OML code is setting a rule how to do the CST filtering function on a feature (spatial attribute). The filtering condition is defined with the OGC Common Query Language (CQL).

4.6 CONCEPTUAL SCHEMA TRANSFORMER (CST)

The conceptual schema transformer (CST) allows mapping of a source dataset to a different target dataset based on the structure of the target schema. It uses Ontology Mapping Language (OML) to make the data transformation [29]. As we have seen in Chapter 3, CST is a WPS that accepts source schema, target schema and source datasets and based on the OML rules produce the target dataset. Humboldt CST, is a standardized web processing service interface that contains three operations based on the OGC specification,

- getCapabilites: returns the service metadata.
- Describeprocess : returns the description of a process or task.
- Execute: performs the actual data transformation.

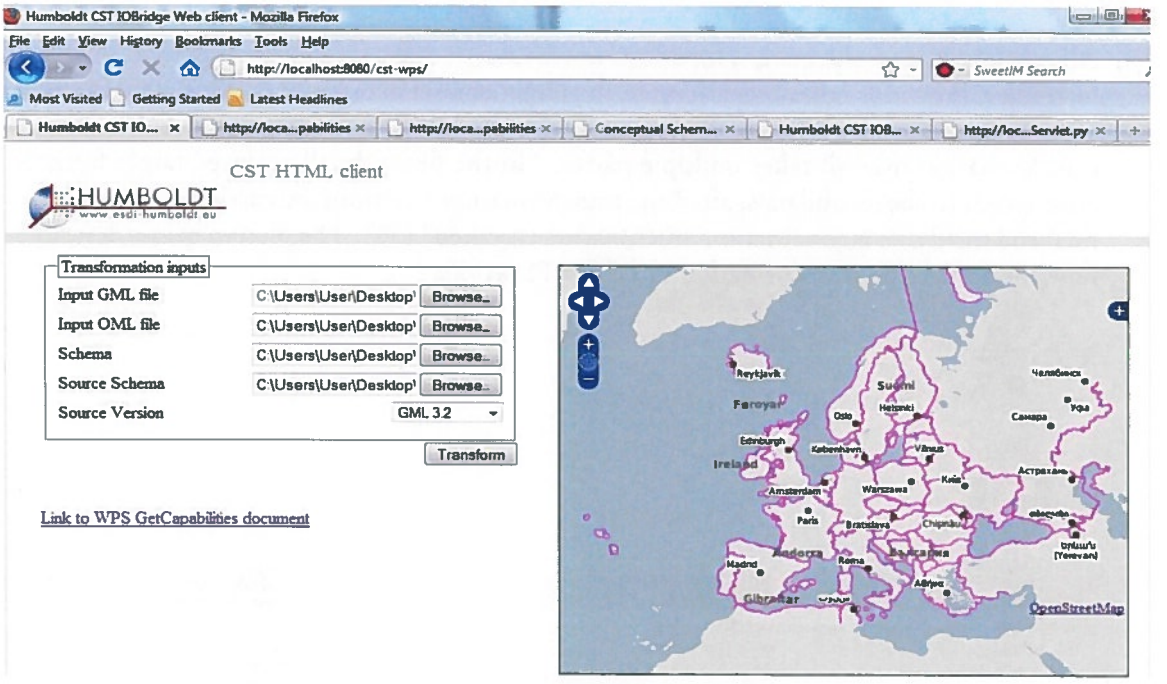


Figure 4.3: CST web processing service

4.7 USE COORDINATE TRANSFORMATION SERVICE (CT)

"The HUMBOLDT Coordinate Transformation Service (CT) is a WPS implementation that allows transformation of coordinates between various geographic reference systems" [5]. If a source

2. The rename schema transformation function that we discussed in Chapter 3, works for any attribute with attribute name a in the source schema and change its name to any attribute name b in the target schema. However the rename attribute function in HALE only works for alphanumeric attributes this means spatial attributes cannot be renamed, instead HALE uses retype feature function. The retype feature function maps a spatial feature into another name or maps spatial attribute from the source schema into another attribute name in the target schema.

3. The rename table that we have discussed in Chapter 3 is not implemented in HALE. The rename table CST function changes the name of the table in the source schema to the target schema table name and maps the values of the source table datasets to the target dataset. This function works with the perquisite that the attribute names, data types and the attribute order of source table must match with the target table.

4. In Chapter 3 we discussed how type conversion function conceptually works for spatial and non-spatial attributes but HALE implements it in a different way. The rename function in HALE not only changes the name of attribute a to attribute name b, but also maps or changes the data type in one step. This means there is no independent type conversion function in HALE. For example the rename function maps from line string to multipoint, polygon to line string, polygon to multipoint and so on.

5. Filtering function that we discussed in Chapter 3 takes the whole table and returns a set of values that meet the requirement. Meanwhile, HALE filtering works on attribute level, the filtering condition returns the attribute values that meets the filtering condition. The attribute filtering could be on spatial attribute or non-spatial attribute but the filtering must be carried out after retype feature function or rename attribute function for spatial and non-spatial attributes respectively.

   If we want to filter the whole table HALE does not provide that functionality. We believe filtering table elements and map them to a target dataset is a necessary function. However, filtering of a table based on a filtering condition need an exact matching of a source table attribute name, attribute order and attribute data type with the target attributes in the target table and this condition is difficult to meet more often but may exist.

Chapter 5
Discussions, Conclusions and Recommendations

This chapter recapitulates the main objective of the research in terms of research questions that we made in Chapter 1 and discus the achieved goals and provides scientific reasons for the goals that are not achieved in Section 5.1. In addition, offer a brief conclusion of the MSc research project in Section 5.2 and make recommendations for further research to facilitate the improvement of the research outcome.

5.1 DISCUSSIONS

The main objective of this research is to design and implement a web processing service for our farm simulation model (FSSIM) in an open service-oriented architecture (SOA). To achieve this objective, three research questions were put forward in Section 1.2.2. In this section we discuss the achievements of our research and the problem we encountered in accordance with the research questions.

1. How to translate simple input from an user to fetch spatial data from different catalogues?

   This research project takes a farm simulation model as a background scenario. A farm simulation model (FSSIM), which is currently being designed in ITC for the estimation of crop productivity plan. It is intended to be used by the farmers of Burkina Faso, to help them decide what type of production mechanism to follow. However, FSSIM requires a diverse type of input parameters to produce its estimation. Some of the input parameters are, the type of fertilizers used, seeds used, number of labor used, the size of the farming land, and so on. This type of input should be provided by the end user of the model, who is a farmer in Burkina Faso but most farmers do not know how to search and provide these datasets. So there should be a mechanism that translates the simple input from the farmer to fetch spatial and non-spatial datasets the model needs.

   All the necessary datasets or almost all the necessary datasets that the model requires can be accessed from different catalogue servers. However, the datasets are in different format, structure, scale and coordinate reference system. Therefore translating simple input from the user to fetch spatial data from the catalogues will create heterogeneous datasets that cannot be used directly in the model. These Heterogeneous datasets need to be harmonized every time a search request is made.

   As we discussed in Chapter 2, harmonizing the heterogeneous datasets that we have requires dealing with the issue of syntactic, semantics and schematic heterogeneity. Syntactic heterogeneity, which has a different type of language to represent the different datasets collected, could be resolved by using OGC Compliant web service since web services use XML-encoding for the input and output [24]. But the issue of semantic heterogeneity is a critical issue still under research. Furthermore, the issue of schematic heterogeneity on web service environment needs the involvement of end or expert user for creation of schema

As we discussed in issue 1 above, we can not make automatic schema transformation on the fly without end user involvement. The end user has to provide the knowledge of which attribute in the source schema is related to which set of information in the target schema. Therefore, the notion of web chaining is not necessary when the objective of automatic schema transformation is unachievable.

Our case study uses the CST-WPS for transforming the different schematic datasets into target datasets with the target schema, as we discussed in Chapter 4. We also deployed a co-ordinate transformation service from HUMBOLDT, a web processing service that accepts a source dataset and returns a dataset with the predefined target reference system. However, it was unnecessary to make the coordinate transformation operation in the case study that we had in Chapter 4 since most of our datasets are non-spatial and the ones that are spatial are in the target reference system.

However, if the issue of automatic schema mapping gets resolved it will be necessary to chain the CST-WPS and the coordinate transformation service that we have. A service that translates a simple user input to a spatial data search and the services (CST-WPS and coordinate transformation service) could all be chained by aggregate service chinning mechanism as we have discussed in Chapter 2. In aggregate service chaining the output of one service will be the input to the other service and it does not need end user involvement. But since the user does not involve in providing the input to the service, the output of the chaining might give back incorrect result [4].

5.2 CONCLUSION

Some of the questions of this research were achieved and some research questions were answered. We propose a schema mapping mechanism to harmonize the heterogeneous datasets available. A mathematical schema transformation concept were developed and tasted with already existing software, HALE (which provides a graphical user interface to produce an OML file) and CST-WPS that provides a schema transformation service using the OML file created by HALE, XML source and target schemas and a GML file as a data encoding standard. A case study was chosen, namely FSSIM farm simulation model that needs diverse input datasets from different data providers.

5.3 RECOMMENDATIONS

The following recommendations were formulated to improve the output of this MSc research project and are discussed as below.

- Implementation of automatic schema mapping service that does not need end user or expert user involvement. The issue of automatic schema mapping requires the devolvement of a mechanism that can resolve semantic heterogeneity automatically. Removing semantic heterogeneity automatically is a topic under research and having accomplished this would also remove a bigger problem for automatic schema mapping since both are interrelated.

- Our second recommendation also depends on the issue of automatic schema mapping; this recommendation should be valuable if the first recommendation can be implemented. That is implementation of a service chinning for our FSSIM model. The chaining should be on;

    - A WPS, which takes a simple input from the end user and returns spatial datasets which meets the model requirement.

LIST OF REFERENCES

[1] W3c ubiquitous web domain. Technical report, W3C Working Group, 12-05-2010.

[2] Trias Aditya and Rob Lemmens. Chaining distributed GIS services. *Altova, Inc. WhitePaper*, 2003.

[3] Koen Aerts, Karel Maesen, and Anton Van Rompaey. A practical example of semantic interoperability of large-scale topographic databases using semantic web technologies. In *Proceedings of the AGILE06, Visegr*, pages 35–42, 2006.

[4] Nadine Alameh. Chaining geographic information web services. *IEEE Internet Computing*, 07(5):22–29, 2003.

[5] Ulrich Schäffler Ines Michi Moses Gone Astrid Fichtinger, Joachim Rix and Thorsten Reitz. Data harmonisation put into practice by the humboldt project. 2010-06-22.

[6] Matt Beare, Simon Payne, and Richard Sunderland. Prototype report for the INSPIRE schema transformation network service,version 3.0. Technical report, September 2010.

[7] Anthony R. Beck, Anthony G. Cohn, Michael Sanderson, Steve Ramage, Chris Tagg, Gaihua Fu, Brandon Bennett, and John G. Stella. UK utility data integration: overcoming schematic heterogeneity. volume 7143, pages 71431Z+. SPIE, 2008.

[8] Yaser Bishr. Overcoming the semantic and other barriers to gis interoperability. *International Journal of Geographical Information Science*, 12(4,299-314), 1998.

[9] Alan Brown, Simon Johnston, and Kevin Kelly. Using service-oriented architecture and component-based development to build web service applications. *CA: Rational Software Corporation*, 2002.

[10] Erin Cavanaugh. Web services: Benefits, challenges, and a unique, visual development solution. Technical report.

[11] John Colgrave and Karsten Januszewski. Using WSDL in a UDDI registry, version 2.0.2 - technical note. Technical report, 2004.

[12] Isabel Cruz and Afsheen Rajendran. Exploring a new approach to the alignment of ontologies, 2003.

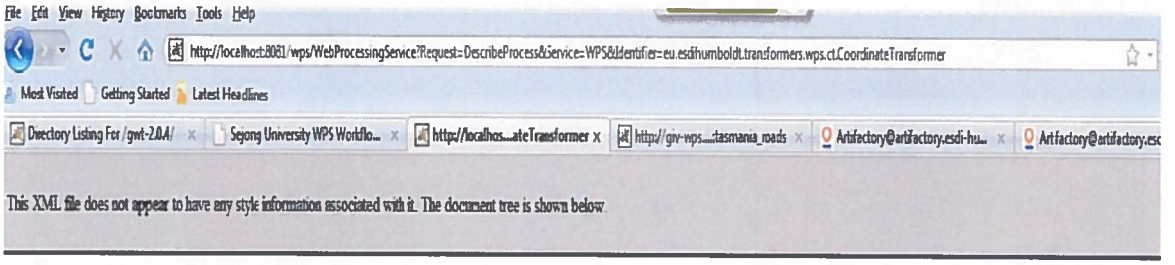
[13] Mike Dean and Guus Schreiber. OWL web ontology language reference. W3C recommendation, W3C, February 2004.

[14] A. Doyle and C. Reed. Introduction to OGC web services. *OGC Interoperability Program White Paper, May*, 2001.

[15] Theodor Foerster, Lassi Lehto, Tapani Sarjakoski, L. Tiina Sarjakoski, and Jantien Stoter. Map generalization and schema transformation of geospatial data combined in a web service context. *Computers, Environment and Urban Systems*, 34(1):79–88, 2010.

Appendix A

HUMBOLDT coordinate transformation service;




```
-<ns:ProcessDescriptions xsi:schemaLocation="http://www.opengis.net/wps/1.0.0 http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd" xml:lang="en-US" service="WPS" version="1.0.0">
  -<ProcessDescription wps:processVersion="1.0.0" statusSupported="false" storeSupported="true">
    -<ows:Identifier>
       eu.esdihumboldt.transformers.wps.ct.CoordinateTransformer
     </ows:Identifier>
     <ows:Title>HUMBOLDT Coordinate Transformation WPS</ows:Title>
    -<ows:Abstract>
       Transform coordinates of a given geometry from one known spatial reference system into another.
     </ows:Abstract>
     <ows:Metadata xlink:title="spatial"/>
     <ows:Metadata xlink:title="geometry"/>
     <ows:Metadata xlink:title="GML"/>
    -<DataInputs>
      -<Input minOccurs="1" maxOccurs="1">
         <ows:Identifier>data</ows:Identifier>
         <ows:Title>Input Geometry</ows:Title>
         <ows:Abstract>The Geometry to transform</ows:Abstract>
        -<ComplexData>
          -<Default>
            -<Format>
               <MimeType>text/XML</MimeType>
              -<Schema>
                 http://schemas.opengis.net/gml/3.1.0/base/feature.xsd
               </Schema>
             </Format>
           </Default>
          -<Supported>
            -<Format>
               <MimeType>text/XML</MimeType>
               <Schema>http://schemas.opengis.net/gml/2.1.2/feature.xsd</Schema>
             </Format>
```


Figure A.1: HUMBOLDT Coordinate transformation service

OML file for transforming from a source schema to a target.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

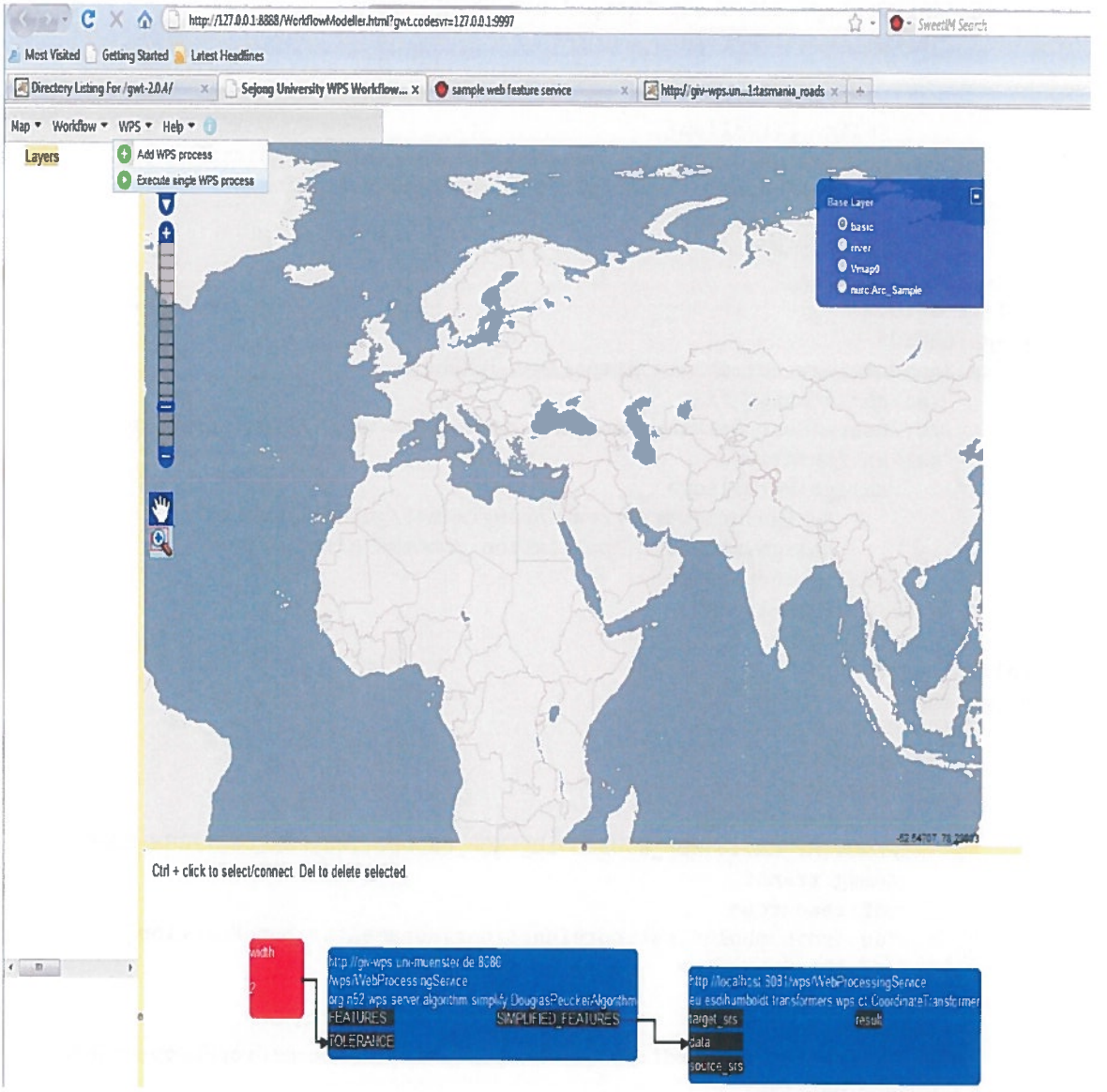


Figure A.3: Example of web chainning

```
            </omwg:entity1>
            <omwg:entity2>
                <omwg:Property rdf:about="http://www.MYURL.com/Village/the_geom"/>
            </omwg:entity2>
            <align:relation>Equivalence</align:relation>
        </align:Cell>
    </align:map>
    <align:Cell>
  <omwg:entity1>
    <omwg:Class rdf:about="http://www.MYURL.com/BF_VILLAGE">
      <omwg:transf rdf:resource="RetypeFeatureFunction" />
      <omwg:attributeValueCondition>
         <omwg:Restriction>
            <goml:cqlStr>LEVEL == 1</goml:cqlStr>
         </omwg:Restriction>
      </omwg:attributeValueCondition>
     </omwg:Class>
     </omwg:entity1>
      <omwg:entity2>
        <omwg:Class rdf:about="http://www.MYURL.com/Village"/>
     </omwg:entity2>
  </align:Cell>
</align:Alignment>
```