Integration of a Web Processing Service (WPS), GIS and hydraulic modelling (TELEMAC) for geophysical analysis

Yicheng Zhang

Department of Engineering, University of Liverpool

September 16, 2014

Publications

Morsali, A., Zhang Y.C., Chen M., Chen, D., 2011. Development of a contaminated land risk assessment model HERA-Soil-GIS in open source QGIS, Environmental Modelling & Software.

Zhang, Y.C., Shams, S., Torres, J.A., Leone, A., Li, M., Chen, D.Y., Integration of Web Processing Service (WPS) and Openlayers in Hydro Information System (HIS) for assessing water balance of a catchment area, Journal of Hydro-Environment Research. (In review)

Contents

List of Figures

List of Tables

Symbols

3O-HIS	Object Oriented (OO) technologies with extensions of OpenGIS standard
DEM	Digital Elevation Models
EPA	Environmental Protection Agency
FWD	Floodline Warnings Direct
GIS	Geographic Information Systems
GML	Geography Markup Language
HIS	Hydrologic Information System
LTER	Long Term Ecological Research Network
NEON	National Ecological Observatory Network
OGC	Open Geospatial Consortium
OO	Object Oriented
SOA	Service-Oriented Architecture
SOAP	Simple Object Application Protocol
WSDL	Web Service Description Language(
WPS	Web Processing Service

Acknowledgements

I would like to express my gratitude and thanks to my supervisor, Dr. Ming Li, whose expertise, understanding, and patience, added considerably to my graduate experience. I appreciate his vast knowledge and skill in supervising me throughout my PhD works and for advising me on TELEMAC and motivating me to use computing knowledge (FORTRAN). I would also like to thank my supervisor Professor Daoyi Chen, who provided his valuable advices at all levels of the research project. I would also like to thank my secondary supervisor Dr. Xiaoxian Zhang for his advice.

I would like to thank my family and friends for their support and for all the great times we've had over the last few years. Finally, I would like to thank Dr. Shahriar Shams for his advice for my

article based on my research works and also motivating me during my stress time.

Abstract

In recent years, environmental management is shifting from a single considerable paradigm of modelling which providing answers to specific queries such as future states of a particular environmental system to much more complicated integrations of environmental applications. For example, determining and visualising coastal zone inundation usually needs to integrate many elements such as population, infrastructure and environment to process an assessment of coastal vulnerability. Because these elements are not independent and interactive with each other, single environmental application can not represent environmental processes comprehensively. This phenomenon commonly exist in environmental field such as population expansion affects environment; the changing weather leads extreme conditions in short term; sediment accumulation and

pollutants potentially damage infrastructure in long term. They are interactive and relate with physical, hydrological and biological processes and needs interoperation among difference organizations such as meteorology, ecology, economy.

In order to fully understanding of environmental process and possible implications to population and environment, an integrated approach is required for encapsulating different hydrological models and a multi-disciplinary management. A typical system presents in this dissertation is the development of the integrated modelling system (3O-HIS) based on the service principle of matching the hydrological data with corresponding web services. The advantage of multi-processer computing is noticeable and a trend that the modern modelling system increasingly makes use of network-linked computing service with information distribution system, instead of the traditional single server based approach. In the modelling system,

commonly available data such as elevation and image data is fetched from web services, processed in hydrological models and results can be analysed as well as Geographic Information Systems (GIS) software. Different hydrological models can be integrated in one system, each model is encapsulated in services and the loose couple feature supports to compose freely for solving a particular environmental problem. Therefore, it can meet the requirements for multi-scale and multi-object assessment and decision-making.

From the precious studies, the multi-object assessment is a goal for many researcher achieved by developed an open standard of hydrologic system. For example, Castronova et al. (2013) developed an OpenMI interface for hydrologic information system; Tarboton et al., (2009) developed a HydroServer which includes an ODM database for managing time series data. However, to achieve these goals is not easy. When dealing with

complicated modelling scenarios, the gathering input data from multidisciplinary sources, unifying data formatting and distributing information across the multi-agency system are still a significant challenge and may hinder the development.

Limited studies have discussed the solutions. This dissertation aims to fill in this gap by developing a new 3O-HIS (Object Oriented (OO) technologies with extensions of OpenGIS standard Hydrologic Information System). The study focuses on the key links to 3O-HIS with hydrodynamic modelling by exploring ways to provide interoperability between services. It streamlines the links of the modelling chain, such as distributed geospatial processing, with services using Open Geospatial Consortium (OGC) standards, Web Service Description Language (WSDL), Geography Markup Language (GML) and Simple Object Application Protocol (SOAP). The platform provides a feasible way to collect hydraulic data from

multidisciplinary data at distributed locations via multiple web services. Hydrodynamic modelling encapsulated in services based on service-oriented architectures (SOA) has the capability in managing, analysing, and publishing images and numerical results, and the integration of 3O-HIS with GIS is convenient to capture, store, manipulate, analyse, manage, and ultimately present all types of geographical spatial data.

Two case studies: flooding simulation at Blackpool in England and water resources management at Demra in Bangladesh are considered in order to demonstrate the advantages of GIS with hydrodynamic modelling using 3O-HIS architectural style. We illustrate the advantages in two aspects: 1) to demonstrate our new 3O-HIS GIS-based approach is better than the hydrodynamic modelling itself and 2) to compare 3O-HIS GIS-based approach with other web services based on GIS.

One unique feature in 3O-HIS is Telemac hydrologic modelling encapsulated on the services and can be run by the processing flows between web services under SOA. The data on multiple services can be interoperation for policy makers, planners and coastal engineers. The considerable assessment between GIS and hydrodynamic results under web services using 3O-HIS indicates a promising future for GIS application in hydrological modelling.

Chapter 1 Introduction

1.1 Background

Computer model has now been used extensively and plays an important role in fulfilling the core tasks of policy making, water resources management and research (STOWA/RIZA, 1999). However, from an application-oriented perspective, Mineter et al. (2003) argued the need for a new generation of environmental applications that from an application-oriented perspective, shifting from the centralized, local desktop applications towards loosely coupled unit which composed of a service interface and a service implementation.

More recently, a number of geospatial studies, involving hydrologic modelling, have illustrated the use of web services

can be very effective to achieve the goal. Alonso et al. (2004) implemented service-oriented architectures and Foster (2005) developed standard interfaces and protocols that allow developers to encapsulate models as services. Lecca et al. (2011) used ideal service-oriented architectures for hydrological applications. The hydrological models and service interfaces for clients are loosely coupled at globally distributed locations. Granell et al. (2009) and Feng et al. (2011) also developed the approach to access geospatial computational models using GIS techniques and Open Geospatial Consortium (OGC)[1], Web Processing Service (WPS)[2] standard. A significant improvement can be found in Castronova et al. (2013) with models being integrated in the framework of the Consortium of Universities

1 OGC: **Open Geospatial Consortium(OGC)**, is an interface standards which has an consortium of 475 companies, government agencies and unicersities participating for publicly available interface standards (OGC,2014).

2 WPS: **Web Processing Service(WPS)**, is an interface standard which provides rules for standardizing how inputs and outputs for geospatial processing services(WPS,2014).

for the Advancement of Hydrologic Science, Inc. (CUAHSI) HIS web services under the OpenMI standard.

Similarly trend and development can also be found in the coastal management and flood prevention. For example, in the UK, it is estimated that 20 times of the risk of flooding is increased due to the climate change according to the Environment Agency government's 'Foresight Future Flooding' report, and it will more than double the number of people at a high risk from flooding (Pitt, 2008). By 2080, flood management investment is required to be increased to maintain the current level of flood risk from £1 billion to £2 billon per annum as estimated by the House of Commons Environment Food and Rural Affairs Committee (House of Commons EFRA Committee, 2008). It is still a challenge to use even the present day state-of-art modelling tool to reduce the cost and to understand the circumstances such as flooding impacts on people, property and

critical infrastructure, and the implication for flood risk management as set out by the Environment Agency, part of the Department for Environment, Food and Rural Affairs (Defra). It has been recognised that the development of a new flood warning system based on a Service-Oriented Architecture (SOA) will be able to deliver timely warnings and improve the performance of the existing flood modelling and hence management capability.

In recent years, SOA has been developing rapidly for helping people manage the water resources and take control of flood risk with innovative service such as sending timely warnings, using multiple communication channels to the public. For example Floodline Warnings Direct (FWD) application has been developed in UK which is based on SOA that enables the Environment Agency to determine target areas for pre-defined warning messages. During the coastal floods of January 2008,

FWD delivered 29,000 calls; and in April, 2012 the system delivered 54,000 warning calls to address 459 areas vulnerable to be flooded, an 86% increase in comparison to previous model. Applications based on SOA are suitable for sharing hydrologic data and facilitate policymakers on future coastal flooding management.

Using SOA, it is convenient for data transmission from user to services or data communication between services. However, it is facing new challenges for developing service-oriented and component-based environmental models. The new concepts and methodology haven't form common 'standards' or 'protocols' to guide us to establish a service-oriented application; in particular for environmental management for multi-scale and multi-objective assessment and decision making. Sharing data across observatories is currently complicated by the fact that the volume of heterogeneous data is explosion and investigated by

disparate research groups. In 3O-HIS， many considerable common protocols such as Simple Object Application Protocol (SOAP)[3] and Web Service Description Language (WSDL)[4] have been applied to integrate heterogeneous types of data and knowledge encapsulation of models as Services. It has the ability of scientists to discover and use data from more than one repository. We listed two problems that it exists when creating a data collection and analysis infrastructure at large scales.

The challenge of developing 3O-HIS is no particular ‘standards’ or ‘protocols’ to define the rules to build a service-oriented application i.e. a model integration platform for environmental applications. Currently, two mature specifications: SOAP and

---

3 SOAP : **Simple Object Access protocol** (**SOAP**) is a protocol specification for exchanging structured information in the implementation of web service in computer networks.

4 WSDL: **Web Services Description Language (WSDL)** is an XML_based interface definition language that is used for describing the functionality offered by a web service.

WSDL used in SOA to form the message-oriented underlay which could have problems in model integration for environmental application. By using SOAP and WSDL, large number of objects (e.g. elements of Mesh) and time series of hydrodynamic features can be transmission between services, it is simplify and concise for the system. The transmission of massive and heterogeneous types of data between services is improved.

Rare hydraulic systems involves ‘Interoperability’ concept which is the feature providing in designing processing. They were generally designed to do a particular job and it is difficult for interoperability when modelling encapsulated as services and is hardly for disparate data resource exchange interactively. This dissertation represents the knowledge that encapsulated hydrodynamic modelling on services to achieve the interoperability that vary researchers can use the model and

disparate data exchange seamlessly.

1.2 Objectives

These purposes that include: 1) understanding of the natural processes 2) testing of the representation of process 3) providing the likely future state of an environmental system 4) consideration and representation of trans-disciplinary systems to support making a decision. And it transmits to the objectives of this research are;

- To examine SOA linking with hydrodynamic modelling to represent multi-objective assessment and decision making.
- To understand the heterogeneous resource communicated efficiently under web services to drive hydrodynamic modelling.
- To understand GIS-based 3O-HIS technology for making decisions based on flood warning maps under different scenarios.

Chapter 2 Services and Service-Oriented Architecture

2.1 Introduction

This chapter is centred on the concept of services and service-oriented architecture to create distributed applications for the research in hydraulic science. To support the modelling integrations, a range of considerable approaches, tools such as Geographic Information Systems (GIS), object-oriented concepts, service-oriented concepts, and component based modelling techniques and modelling frameworks.

In recent years, environmental management is shifting from a single considerable paradigm of modelling for multi-task modelling which providing answers to specific queries such as

future states of a particular environmental system to much more complicated integrations of environmental applications. This chapter provides an overview of the existing approach of integrated applications from the conceptual and technological aspects. It covers a review of the developments in recent years including service-oriented concepts, component-based modelling techniques and modelling frameworks. Solutions are reviewed which are migrated from dedicated desktop to on-line, loosely coupled and standards-based services in order to meet the requirements for multi-scale and multi-object assessment and decision- making.

2.2 Overview of system architecture

Over time, the level of abstraction at which functionality is consumed has gradually become higher and higher. It has progressed from models, to objects, to components and even to services (Sprott and Wilkes, 2004). Recently the system

architecture including Service-Oriented (SO) concept has been discussed frequently which a computing integrated services and a computing paradigm. SO has been developed that utilizes services as fundamental elements for changing mission requirements. Major advantages of Service-Oriented Architectures (SOAs) are 1) a loosely coupled architecture which can support dynamic simulation process; 2) a distributed system architecture for data sharing in an Internet-based infrastructure (Georgakopoulos and Papazoglou, 2012; Peckham and Goodall, 2013); 3) services are suitable to encapsulate software capabilities (Sprott and Wilkes, 2004), and 4) higher-level abstractions are provided for organizing applications for large-scale and open environments (Granell et al., 2009).

Based on SOA, a range of environmental modelling and software development approaches encapsulated services for

supporting and increasing management requirements dealing with huge data resource in the environmental science fields. (Georgakopoulos and Papazoglou, 2012; Argent, 2004; Papajorgji et al., 2004). From a range of environmental modelling, the trend of development of system architectures is from centralized, object-oriented integration platforms through to the loosely coupled, service-oriented programs, the mission requirements are modified to focus on between services. Interoperation between services is important that services distribute seamlessly to eliminate barriers. The integrated modelling platform abstracts the functionality in a high level and provides service implementation independently and interactively for multi-objective assessment.

2.3 Object-oriented and Service-oriented Paradigm

Object-oriented paradigm (OO) and service-oriented (SO) paradigm are the two commonly system architectures used to

encapsulate functionality for the purpose of interoperation in the integrated hydraulic applications. OO paradigm defined the function usually recalled on a local machine while SO functionality is provided by procedures of a service. They both can encapsulate the hydraulic models with objects or class. For example an OO approach is explained by Guariso et al. (1996) that a strong integration of the various modules implemented under OO structure and Rizzoli et al. (1998) developed framework–based approaches using OO structures for optimization, scenario exploration and decision support. Recently, Argent (2005) developed a simple catchment-based nonpoint source pollution models using OO paradigm as modelling methods, and different data types in this system architecture to estimate pollution loads.

2.3.1 Existing configurations

Overview of OO paradigm, it reveals that OO structures support

well the concepts of model reuse and extensibility. It encouraged users to define their own hierarchy of model classes. OO structures can separate the different modelling as components and such distributed mechanism ensure that one modelling changed do not call for changes in others. However, OO structure is not beneficial when thinking about requirement of the light of existing models constraints (Bih, 2006), especially isolated service requests/components. On the database management side, OO framework almost inevitably becomes much more complicated because it stores massive objects for representation of processes such as hydraulic tide process. There is a need for large memory for storage in databases and even duplicate the data.

While SO paradigm solves these issues and satisfies the new requirements in light of constraints such as decoupled and independent service implementation. The principle of “services

and messages" in SO architecture (SOA) provides an attractive way for integrating different modelling, especially design a complex framework. Services can store different types of data and data sets can be communicated by messages. For example various and massive hydraulic attributes such as bathymetric information, digital elevation models' (DEM) maps, water depth and velocity are arranged in difference resource repositories (NASA, USGS, OSGE, EPA) and are interacted by messaging under different protocols (SOPA, WSDL).

SO paradigm concerns interoperation between modelling have to fulfil an interface for linking other modelling. SO structure can fulfil the emerging requirements to flexibility operation flow because SO structure is essentially a collection of services and the communicating and connecting services to each other remotely. It creates a flexible framework. The challenge is the scale of framework need to be carefully designed using SO

paradigm for researchers to develop models that operate Level I-IV which is explained in details in model integration by Argent (2004).

Compare with OO paradigm, SOA can generally represent any data independently to encourage reuse of data and other resources for different users' scenarios. SO paradigm is more attractive and interactivity provides different distributed computational components and implies the superiority that distributed components interoperate over a network (Huhns and Singh, 2005). The following section shows how the principles of SOA are embodied by modern Web Services practices.

2.3.2 Embracing SOA structure

A design strategy for large information systems, service-oriented architecture (SOA) is an evolutionary distributed computing from the design hypothesis based on requests and responses for

synchronous and asynchronous applications and relies on a collection of loosely coupled self-contained functional modules, referred to as services, which communicate with each other and can be called from multiple clients or chained in processing workflows. SOA includes comprised of three primary parties: Provider (of services), Consumer (of services), Directory (of services), as shown in Figure 2.1(Huhns and Singh, 2005). Providers publish or announce their services on registries, where consumers find and then invoke them. These services have free connection that service interface is independent implementation.

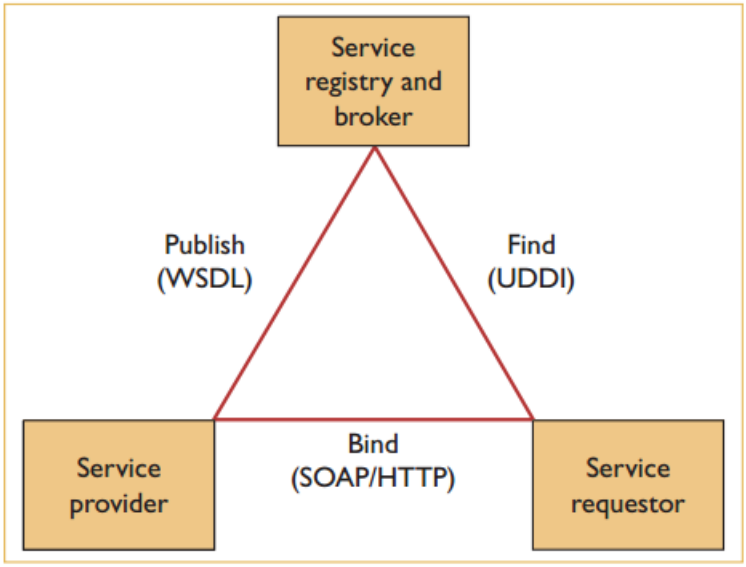


Figure 2.1 SOA explained (Huhns and Singh, 2005, After Service-Oriented Computing: Key Concepts and Principles, Page 76).

The service provider is the network-addressable entity that accepts and executes requests from consumers (Chatterjee, 2004). It is the mainframe of the architecture, components executes the service request. Service registry contains available services which stores contracts from service providers and provide to the service consumers. Usually service registry used to identify services, describing the service and the service function. Service requestor requires services for achieving an idea or a solution. Service requestor nodes often discover and invoke other software services that need call remotely to a distributed object, the service provider.

In SOA there is a fundamental abstraction from which all higher-level functionality is built: messages. It supports the requestor/provider services communicating in an interoperable way. A canonical service may resemble that of Figure 2.2. It contains the layer of messaging, logic, and resources. The

services communicate with each other's only in message processing layer which represent an "event horizon" (Webber and Parastatidis, 2009). Message exchanges between services where the messages arrive and leave in message processing layer. It resemble interface/function application, the internal architecture of a service is kept hidden from the outside world.

SOA provides the standard communication protocol as well as methodology to discover, invoke, and publish online web services (Horsburgh et al., 2009). The protocol view defines the message formats and message exchange patterns. Some principles need to follow for the design of protocols: boundaries are explicit; services are autonomous; services share only schema and contract; and policies determine service compatibility (Webber and Parastatidis, 2009). The message functionality is a fundamental abstraction that makes transmission between services. Therefore, message exchange

between services makes the chance for designing services independently. For example UK hydraulic data (e.g. water depth) stored in National Oceanography Centre service, we can obtain the data by using message to contact service registry, and service is processed independently without affecting other services such as bathymetric map service and elevation service.

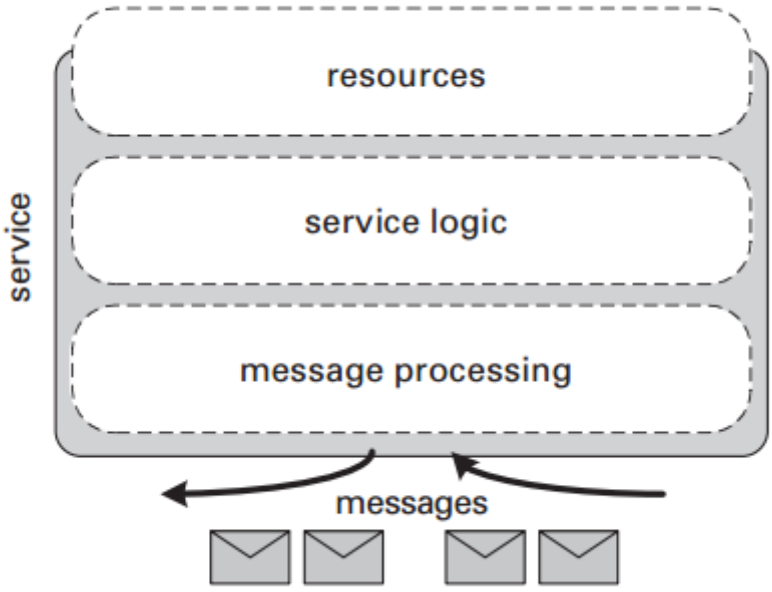


Figure 2.2 The anatomy of a service (Webber and Parastatidis, 2009, after: Service-oriented computing, Realizing Service-Oriented Architectures with Web Services, Page 57).

The implementation level focuses on the implementation of flow that services chained in processing workflows. Services are

designed to facilitate the message exchange for the protocols it supports. From many articles, it is important to understand that services should not be considered “containers” which contains serialized objects, functions. It is wrong to relegate them to mere interfacing or encapsulating them as interfacing application. We need to pay less attention on how the functions are “hidden” well in services; we need to consider how a series of services are designed by reading the messages that service application abstracts. With a summary, the loose-coupled services, messages and standard communication protocols provides reusable components and interoperable communications in 3O-HIS.

2.4 Web Services

Web services as a real implementation technology under SOA has been examined including a typical Web Service is designed, implemented, and hosted in a runtime environment. Section 2.4.1 introduces the web services architecture; Section 2.4.2

explains specifications and principles when Web services implement in distributed systems.

2.4.1 The Web Services Architecture

In the early days of the Internet, we shared information efficiently using static websites called HTML. However, this process was not suitable for delivering services over the Internet. The original HTTP and HTML protocol stack often provide static, or at least highly cacheable. In contrast, the Web services architecture is designed for highly dynamic program-to program interactions. The architecture view encompasses web services protocols which are designed with layers cleanly.

2.4.2 Protocol specification

A number of protocols may be implemented in distributed systems such as protocols in Security (WS-Security, WS-Trust), Reliable messaging (WS-ReliableMessaging) and Transactions (WS-AT/BA, WS-TransactionManagement). Three major

principles: message orientation, protocol compatibility and autonomous services have been discussed in the following section.

Message orientation

Message orientation is the core principle that communicates between services using only messages. Three general protocols a WSDL protocol, a SOAP protocol and XML for message communication provide data accessibility over Internet as shown in Figure.2.3.

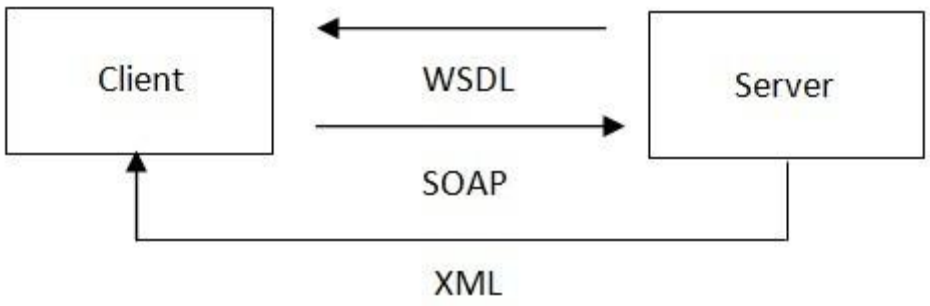


Figure 2.3 General Web services architecture.

It is important to understand how individual messages are formed and processed between services. SOAP is an XML-based, stateless, one-way message exchange protocol for

interacting with web services over HTTP. WSDL is an XML-based format for describing Web services as collections of network endpoints or ports (Lutz et al., 2009; Paul and Ghosh, 2008). The client side receives WSDL files from the server side that learns the service‘s methods, and a SOAP request is then sent to the server to identify the needs. The server processes the SOAP request and sends output in XML format. Unlike RPC systems in which messages are strictly subordinate to the local programming experience, the Web services architecture is built with messages as the atomic unit of communication (Webber and Parastatidis, 2009).

Protocol composability

Protocols can be used either independently or in combination. Based on SOAP as the lowest layer in the protocol stack, new protocols can be composited through the use of a flexible header mechanism. The protocols have two layers: header and body for

message exchanges between services. One advantage is that there is absolutely no cost to applications that only loaded the protocols that used. For a particular application, the external protocols can be used such as WS-Security, WS-ReliableMessaging (Webber and Parastatidis, 2009). This can maximize the applicability of the architecture.

Autonomous services

Services are designed and evolve independently from each other. If there is an internal changing in one service and will not affect other services. Evolution of a service's implementation may not consider the other's behaviour of the service. This feature helps to improve the portability of architectures.

2.5 Key Feature of Web service: loose-coupling

In this section loose-coupling feature is the key to Web services interoperability. It provides a robust way to evolve a service's implementation. Generally, there is another approach to called

tight-coupling integration approach which differs from many of the service-oriented approaches for integrating water resource simulation models. The two approaches tight-coupling approach in Figure 2.4(a) and loose-coupling approach in Figure 2.4(b), Figure 2.4(a) is demonstrated the originally independent models are integrated by porting code into a single modelling application (Band et al., 1993; Facchi et al., 2004; Maxwell and Miller, 2005). The tight-coupling approach provides complete control and complicated numerical problems can be solved efficiently (Pingali and Stodghill, 2004). However, the inflexible architecture is difficult to provide an efficient and reliable manner and hard to ensure the science being performed is relevant to a particular environmental management context. Internal conventions such as data structures and semantics within a model are fixed and become difficult to integrate models. It is not easy to integrate external models which do not define with internal conventions.

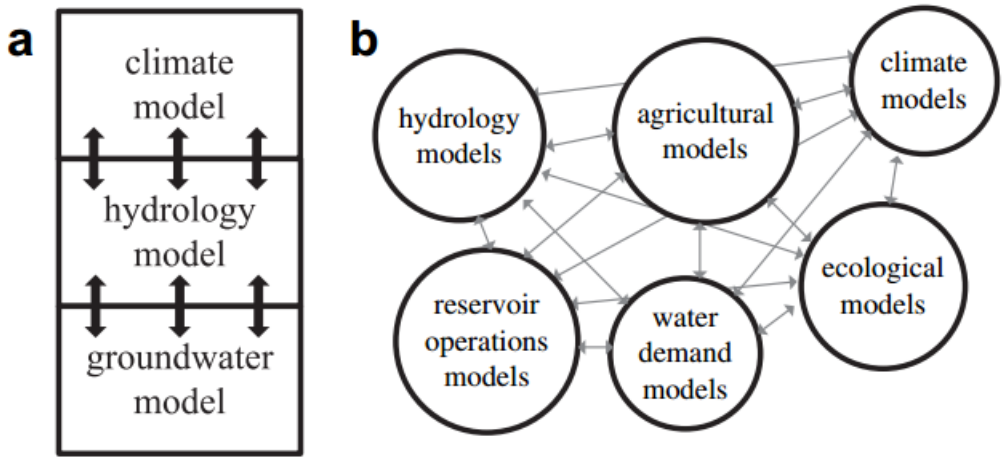


Figure 2.4 Contrasting a (a) tight-coupling and (b) loose-coupling approach for model integration (after: Jonathan et al.,2011, Modelling water resource systems using a service-oriented computing paradigm, Page 574).

A standardization of interfaces is requirement data for development a loosely-coupled architecture. The advantages of adopting a loose-coupling, service-oriented paradigm for modelling are to enhance the interoperation between services and to provide a uniform framework for data exchange. Loose-coupling 3O-HIS will be described in next chapter.

2.6 Summary

The concept of services and service-oriented to create distributed applications for the research in hydraulic science

have been reviewed in this chapter. SOA provides a delivering information structure from hydrologic data in organizations. Under SOA, it is desired to establish a robust system to support data/model interoperation and data publication in distributed networks. CUAHSI HIS has many advantages in accessing disparate information sources, publishing hydraulic data and interfacing with a variety of community models. However the successful CUAHSI HIS has become overloaded, as massive hydraulic data linking with CUAHSI HIS central, the inconsistency of CUAHSI HIS central may occur because vast ontologies need to be paired with hydrologic data stored in web services.

Chapter 3 Methodology

3.1 Introduction

This chapter presents the physical and mechanical concepts of 3O-HIS platform and the methodology of encapsulating hydrodynamic modelling (e.g. Telemac) in 3O-HIS. SOA has been reviewed previously as a basic work frame for developing 3O-HIS. In this chapter, the encapsulating methodology of 3O-HIS has been discussed, including its mechanical concepts, the structure and behaviour of solutions, as well as an example of encapsulating Telemac in 3O-HIS which helps us understand encapsulating hydrodynamic modelling in technological aspects necessarily.

With the background of HIS management, institutional

repositories and scientific process changes as mentioned in previous chapters, 3O-HIS has been developed with the idea of CUAHSI HIS and cyberinfrastructure and supports web service based SOA. It can be seen that it is not only a framework that is better aligned and interact with hydraulic data which extends well beyond just the representation of natural phenomena, but also accommodate new concept applied for multi-scale and multi-objective assessment and decision-making. The existence of huge information repositories have their own framework and cannot be interactive apparently between the external and internal parts. 3O-HIS has been developed for solving this problem and information can be interoperate. This chapter provides a comprehensive method for creating 3O-HIS architecture based on environmental management imperatives underlying recent HIS development, highlights some of the specifications and protocols, semantic, operational process requirements, these are addressed in technological aspects of

integrated modelling to develop 3O-HIS for preparing assessment in an interdisciplinary way.

3.2 Related works

Past work using SOA in the application of web services within the water resources community has focused primarily on exposing historical databases (Goodall et al., 2008), integrating water data across heterogeneous data providers (Horsburgh et al., 2009), or data processing workflows using web services (Granell et al., 2010). Recent attention has focused on service-oriented architectures as a means for building environmental decision support systems (Mineter et al., 2003; Granell et al., 2011; Goodall et al., 2008; Horsburgh et al., 2009). A software system is viewed as independent components or services that are loosely-coupled and able to exchange hydrologic data with one another over a network (Curbera et al., 2002; Huhns and Singh, 2005). In hydrologic fields,

service-orientation is a core concept behind distributed computing where the Internet is used for primarily spatial data analysis. For example integrative Telemac modelling with GIS in the design of SOA works, possibly with the aid of numerical predictions of the coastal morph dynamics and the flood risk analysis of the lowland such as Blackpool in England.

Recently the development of CUAHSI HIS has shown two fundamental aspects: 1) to integrate hydrologic observational data from heterogeneous federal, state, and local data providers into a web service using a service-oriented architecture; 2) to encapsulate hydrologic modelling and modelling interactive aid management decision making (Maidment, 2008; Castronova et al., 2013 and Al-Sabhan et al., 2003). Granell et al. (2010) demonstrate how the role of services can be expanded to handle data delivery, data processing and visualization. Summarize these literatures along the lines given above; the authors use

hydrologic simulation models as part of their service-oriented architecture for multi-scale and multi-object assessment and decision-making.

3.3 Overall design

Design of environmental modelling system undertaken for a vast range of reasons, one motivation of developing 3O-HIS is better understanding the processing of natural phenomenon. The goal of design 3O-HIS is the concept of reuse and interoperation between services; each layer is explicit and each component is processing separately and independently for operation. 3O-HIS has been derived from CUAHSI HIS.

3.4 CUAHSI Hydrologic Information System (HIS)

CUAHSI HIS initiatively is supported by a 5-year grant from the National Science Foundation (NSF) and developed well with a joint effort among more than 100 universities and funded by NSF to advance research in hydrology (University of California

San Diego, 2008). Three key parts: several data storage, delivery and discovery tools and standards have been developed within the context of the CUAHSI HIS project (Tarboton et al., 2009).

Meanwhile, a new comprehensive infrastructure known as “cyberinfrastructure” has allowed the research teams to share distributed data resources via high-speed networks. Cyberinfrastructure has been developed by many activities and organisations to support the needs of multidisciplinary collaborative research, such as the National Ecological Observatory Network (NEON) (Taylor, 2013), the Long Term Ecological Research Network (LTER) (Robertson et al., 2007) and the Geosciences Network (GEON). It has been developed as a part of a HIS that integrates data sources and functions using web services in a distributed network and enables the infrastructure to access, organise and analyse datasets (Horsburgh et al., 2009). Under cyberinfrastructure, CUAHSI

HIS was developed as a part of advanced hydrologic research. Users can implement web services to discover and access a variety of hydrologic data sources and develop applications for the desktop and the web (Zaslavsky, et al., 2007). The goal of CUAHSI HIS is to promote data sharing and reuse through the use of standards and web services for information exchange (Horsburgh et al., 2009).

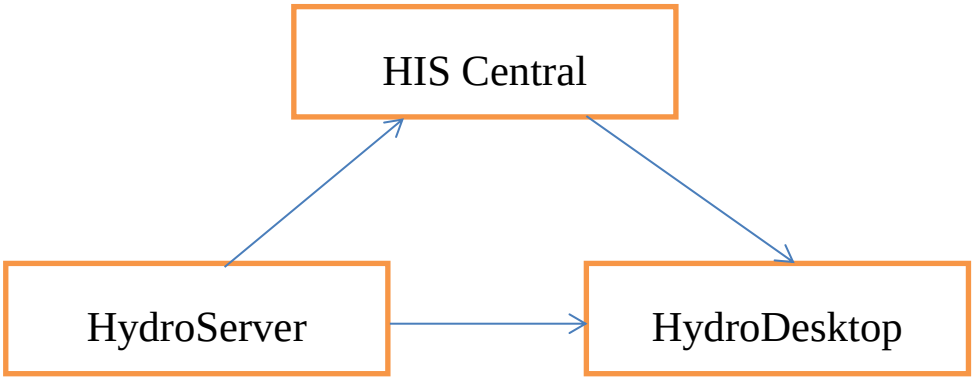


Figure 3.1 The key components of the CUAHSI HIS service-oriented architecture (after: Zaslavsky and Maidment, 2011)

From the Figure 3.1, three classes of functionality compose the CUAHSI HIS services oriented architecture: 1) data collection (HydroCatalog); 2) data transmission, including Observations Data Model (ODM), WaterOneFlow web services and WaterML;

and 3) data publication including HydroServer and representation tools Hydroseek and HydroDesktop. CUAHSI HIS integrated these functions and have the capacity of interoperation of multiple hydrologic and environmental observation data repositories.

The core of the CUAHSI HIS is a WaterOneFlow SOAP service that creates a standard mechanism for interoperating of hydrologic data between servers' and users' computers (CUAHSI HIS, 2010) by providing uniform access to disparate hydrologic observation data from multiple heterogeneous repositories (Zaslavsky, 2007). The WaterOneFlow web services generate XML message named CUAHSI WaterML, which is an XML schema that defines the format of hydrologic data values and time series as well as the hydrologic metadata (Zaslavsky et al., 2007). CUAHSI HIS has also been developed as an Observations Data Model (ODM) that provides a standard

database schema for use in the storage of hydrologic observations data collected by multiple investigators. Within it, a relational database is created in a geographically distributed network to effectively share hydrologic information between investigators.

For the purpose of moving from CUAHSI HIS to 3O-HIS, it is necessary to understand the concept of CUAHSI HIS. It is possible to access disparate information sources in a uniform standards-based manner; easily publish locally collected observational data; and easily interface with a variety of community models and analysis and visualization codes. Meanwhile CUAHSI HIS supports the interdisciplinary research such as a range of climate models related meteorology enable cross-scale analysis of hydrologic cycles. A higher level of interoperability of hydrologic information may save considerable time and money which perform similar task in

multiple environments in particularly data-intensive hydrology research (Shams and Huang, 2009).

Under CUAHSI HIS framework, difference of hydrologic information data such as water properties, distribution and circulation on the earth's surface and high-resolution remote sensing can be retrieved from multidisciplinary repositories at distributed locations (Goodchild, 2007). The disparate data resource is collected through a multitude of agencies including the Environmental Protection Agency (EPA), U.S. Geological Survey (USGS), and the National Oceanic and Atmospheric Administration (NOAA). However, some limitations exist when vast ontologies stored in CUAHSI HIS.

Although disparate data sources can be collected from EPA, USGS, USDA and NOAA, there could have integration issues that not all these databases are incompatible with each other

(Zaslavsky et al., 2007). It means that it is possible to collect hydraulic data from multidisciplinary sources at distributed locations via multiple web services. However, there is a lack of an effective and efficient method to extract expected information from massive and complex spatial databases. For example, it hinders the decision makers to manage water efficiently under the availability of partial databases.

The key issues are the hydrologic community ontology in the HIS central. Although CUAHSI HIS central is a unique feature of the CUAHSI HIS that showed us that it was possible to bridge the gaps to access data across many independently managed observation networks and datasets. However the tremendous success of HIS has resulted in problems of framework inflexibility and became bloated due to vast ontologies that need to be paired with hydrologic data stored in web services. Thus the key issue is linking data effectively and

reliably from multiple services. Additionally, maintaining the central metadata catalogue in synchronization with multiple sources of hydrologic data has been one of the main bottlenecks to CUAHSI HIS. In addition, CUAHSI HIS has not yet developed the capability to integrate other data such as economic, social, land use and remote sensing data.

3.5 3O-HIS architecture

We developed 3O-HIS which was first built by Leone et al. (2006); Spanou and Chen (2000); Leone and Chen (2007); Shams and Huang (2009), using object oriented (OO) technologies with extensions of OpenGIS standard. It inherits the idea of CUAHSI HIS and cyberinfrastructure and supports SOA providing a framework for integration of different types of disparate data, information management, analysis and modelling implemented across multidisciplinary boundaries (Zaslavsky et al., 2007, Chen et al., 2010). 3O-HIS developed based on the

ideas of SOA and the architecture of the system along with different layers including data layer, service layers and presentation layer. The architecture of 3O-HIS has been shown in Figure 3.2.

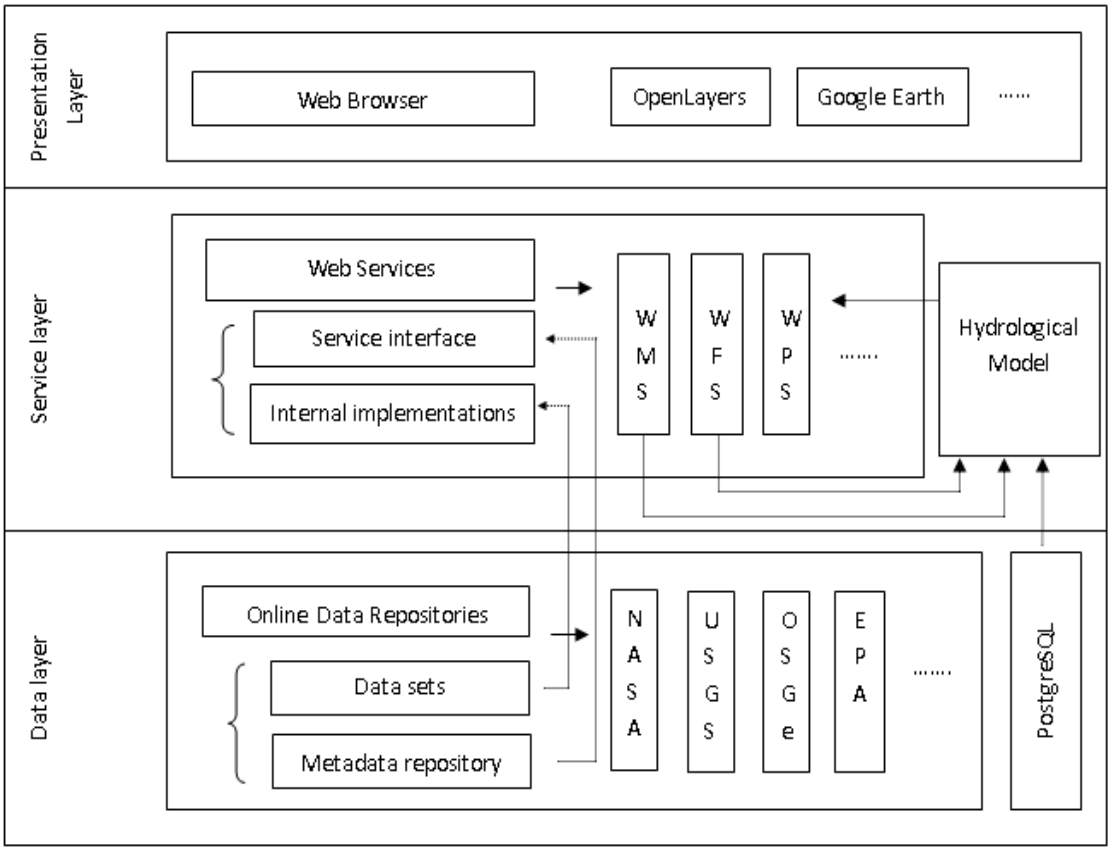


Figure 3.2 The 3O-HIS's architecture

3.5.1 Data layer

Hydrologic data consists of many different kinds of observations such as rainfall, stream flow, and water depth. Federally

organized, observational data repositories, including the above mentioned NASA, USGS NWIS, EPA STORET, USDA SNOTEL, and NCDC provide the hydrology data that are both spatially and temporally distributed. Two major hydrological data are point (discharge) and distributed (rainfall, runoff, evaporation) types as used in hydrological models (Leone et al., 2006).

Researchers in hydrology rely to a large extent on federally organized; observational data repositories. In data layer, different data repositories have similar standard for hydrologic data transmission. For example, the repositories of USGS and NWIS are developing the HydroML language which is similar with WaterML developed by CUAHSI HIS. From the left of data layer, the distribution of hydrologic data can be broken down into two parts, first part is the data sets which mean the data itself and the second one is metadata repository which is

the description of the data being served and repositories. The key value pair metadata is stored in metadata repository and in this way it can be registered to incorporate new data formats. The main purpose of the data layer is to store the metadata of data files and to provide a mapping between hydrological terms and the datasets being served. We also developed a local database frame, called PostgreSQL shown on the right of Figure 3.2. It provides the capability to store results calculated by numerical modelling. This function is an important feature for data collection. For example, if a place that is rarely observed and it lacks of historical or observational data stored in repositories, simulation of natural phenomenon by hydraulic modelling is a considerable way. Results stored in the local system and posted, submitted and shared with other users; provide us a new way for multi-scale and multi-objective assessment and decision-making. The logical layout of the data layer structure of 3O-HIS is shown in Figure 3.3.

Three modules are processed separately and independently for operation. The initial result is calculated in the first part, and GIS can handle spatial data analysis and visualize based on products in the data management block. In addition, the products from GIS can resend to the environmental modelling as a new input file. Based on the results produced by GIS, many likely future states of environmental processes can be represented by re-sending different scenario files.

Figure 3.3 The bottom structure of 3O-HIS

3.5.2 Service layer

Recent development of environmental models is designed typically by an entrenched linear process. In environmental

modelling side, changing other environmental models is not as easy as we expect. To choose other environmental modelling or other database software, the workflow was stopped possibly by the inconsistent methods of connection and the lack of each block linkage. Argent (2005) stated environmental models are rarely designed to communicate with other models within standard disciplines. For example, environmental modelling and data management have different sub environmental system. There is a limited open and extensible architecture for integrated GIS with environmental modelling. The technical limitations and issues lead users to have limited choice but to deal with other developer workflows. For example, users cannot choose a suitable environmental modelling for customized design. It raises a concern as how to harmonise between various environmental tools, without changing sub environmental system. 3O-HIS design has a set of concepts of flexibility, reuse and interoperability.

In service layer, web services application in software engineering is considered as fundamental and vital part in 3O-HIS. It supports the requirements of interaction of information within the organization and metadata information ensures the hydrologic data delivering from multi-repositories to user platform. A web service is a software system that is specified by URI (Universal Resource Identifier) and their general interface is defined and interpreted using XML; other systems can interact with web service by a determined behaviour and in accordance with its definition. Web services are among the most important recent developments in software field which will certainly have remarkable results for designing and implementing applications and paving the way for the new generation of web-based applications. Web services can be used for implementing SOA and create the constructive blocks of the function which are accessible through the Internet. The

designers of SOA typically use web service standards, used extensively in industrial and commercial contexts, to implement their services. These standards remarkably increase the interoperability of commercial systems.

Web service matches the ontological differences and provides the hypothesis based request/response for synchronous and asynchronous applications. We not only need to consider the information from existing repositories, but also design web services for sharing the result stored locally. We show that web services can be used as an initial block in program creation according to several indicators like accessibility, quality, cost and efficiency. After reviewing the general Web services architecture and analysis of required services for organization, we present our 3O HIS service oriented design method.

Key of these demands that influenced by the needs of today's

scientists is “integrated” modelling. That is means different components of the natural other systems are modelled in a linked way representing features of system behaviour. The software architecture of server-client linkage is shown in Figure 3.4. In our designed Web service, XML is generated to solve this challenge, which ensures that the geospatial data interoperate seamlessly between web services.

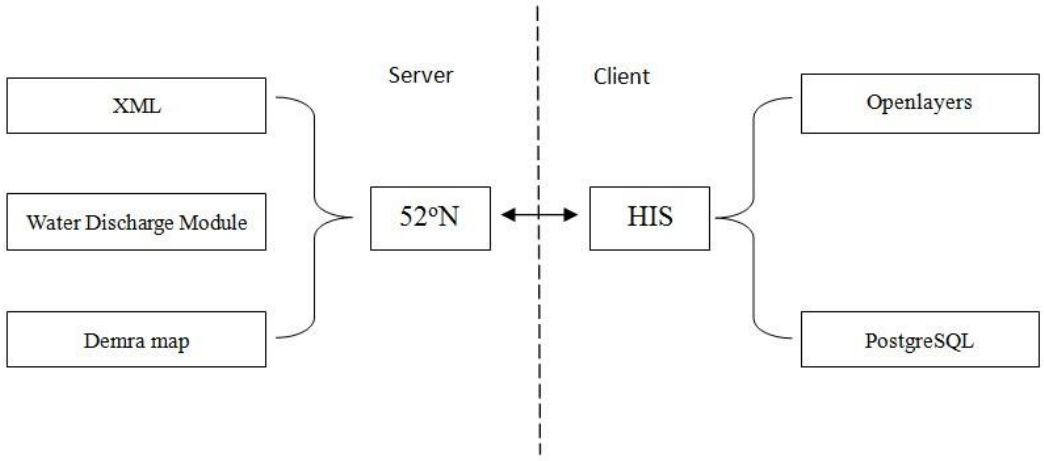


Figure 3.4 52 °North WPS service and HIS

Generally, the Water Discharge Module remains at the server side, while 3O-HIS remains at the client side. At the server side, an XML file defines the DescribeProcess operation. It sends the

type of data for input and output to the service. The Water Discharge Module compiled in Java is uploaded to 52°North WPS. 3O-HIS provides these standard programming paradigms that allows hydrologic modelling as components, located on any network to be discovered, published, and invoked by other modelling web service. Regardless of location of distributed network and development platform is one of contributions using 3O-HIS that any service-based application can interact with any other service-based application based on distributed components.

The first step in creating a service component-based system is the identification of individual components of layers. Identification of each component is not easy working, although there are various hydraulic models used for representation the nature of process and thousands of hydraulic information uploaded by ocean stations, federations and web water

repositories. Most of hydraulic modelling were generally designed to do a particular job and are often not able to provide the features we now desire, such as flexible exchange of algorithms or connect with other models.

3.5.3 Presentation layer

Currently, HydroDesktop software has been developed by Ames et al. (2012). There are many advantages for using HydroDesktop. It accessed hydrologic data services that have been published using CUAHSI HIS and the default of search across all WaterOneFlow web services registered at HIS Central. It can be used for discovering and accessing hydrologic and climate data in different data repositories. However, a difficult task for HydroDesktop consumes valuable time. Two tasks are implemented to reduce time consumption in our designed web service. Zaslavsky et al. (2007) stated for the time issues, limitations to the number and size of hydrologic data stored on

individual workstations, and HydroDesktop platform is missing the ability to query over arbitrary hydrologic themes. Our design system have has the ability to query certain volume of available data using OGC and GML standards and the process time is limited to gather all of the actual time series data values for larger themes from distributed data services. Web services registered at the HIS Central is unrestrained. All web services are interoperable, sustainable, and extensible by using OGC and GML standards for data exchange and community semantics. These standards provide standardised ways of manipulating the GIS data so that data can be utilised more efficiently (Shams et al., 2010; Chen et al., 2010).

In an organization, information of the operation process must be kept on service-receiver or customer side in order to minimize the storage volume of the information and to enhance the speed of the organization. Time issue should be noticed that when the

number of modelling units per simulation is tested in the OpenMI system, the execution time may increase as a nonlinear trend may occur (Castronova et al., 2013). We summarized the result of the OpenMI model execution time stated by Castronova et al. (2013) in Table 3.1.

Table 3.1 The OpenMI model execution time varies as the number of modelling elements

Number of Model Elements (10^3)	Execution Time (Sec)
0-1	0-1
1-2	~1.1
2-3	~1.3
3-5	~1.5
5-7	~1.5

Table 3.1 is the result of the large-scale loading test and illustrates how performance when the number of model elements extends into the thousands in the OpenMI model scales.

The execution time is under 1.5 second when numbers of Model elements below 7000. There is no evidence of nonlinearity in the

OpenMI communication itself. The component-to-component communication represents only a small fraction of the total model execution time. However, model execution time varies with different models as shown in Table 3.2 Choosing an appropriate model as component is important beyond the range of the previous test (Castronova et al., 2013). From the table 3.2 we found different components' execution time varies. This is the major reason for time consumption. One advantage of this type of model is that the components are isolated and we can handle the computational bottlenecks by tracking the runtime of each component. Because each component is loosely coupled within the system, efforts can be made to address computational inefficiencies of a single component in isolation of the larger system. However, Khanbabaei and Asadi (2011) stated loose coupling of services in negative side. The coupling of several services together lowers the efficiency of the web. The occurrence of any problem with each one of these services

leads to the breakdown of the whole application. There is a requirement that increases the accessibility of control of the web to prevent these breakdowns.

Table 3.2 Different components' execution time varies as the number of modelling elements

components	Number of Model Elements(10^3)	Execution Time(Sec)
Initialize Unit Hydrograph	0-1	0-1
	1-2	~2
	2-3	~4
	3-5	~8
	5-7	~9
Muskingum PerformTimeStep	0-1	~17
	1-2	~200
	2-3	~400
	3-5	~1000
	5-7	~2000
Curve Number Finish methods	0-1	~5
	1-2	~10
	2-3	~20
	3-5	~30
	5-7	~40

At first, we attempted to reduce the time required to download reused data, such as those downloaded from the Hybrid maps in the VirtualEarth WMS service. Once reused data are downloaded via a web service, they are accessed as local service

for further needs. The constant values are stored in a local database to increase performance. The land-use map for the catchment or the study area is obtained as GIS data, vegetation cover is calculated using GIS spatial function, and all data are stored in a local database. The catchment area and the vegetation cover, which serve as the inputs for the water balance, can be stored in a local service. The ESRI shapefile format files for the study area hydrological domain are uploaded in ArcMap 10.1, including agricultural, industrial and residential maps. Because the types of crops determine the water storage, as described in section 2, a ‘Reclass’ function was used in ArcMap based on the different types of crops. Subsequently, each sub-area can be calculated in an Attribute Table using a ‘Calculate Geometry’ function. The number of pixels for each type of vegetation cover is determined. Each pixel has a unique ET that is calculated based on a particular type of vegetation with a known crop coefficient. Finally, these values are stored in the PostgreSQL

database as a part of water balance model input. The water balance model is programmed in Java; the model obtains the required data from the remote services and the local database and computes the water storage to compare it with the measured discharge in order to validate the results.

3.5.4 Specifications and protocols for supporting the design of 3O-HIS

The service-oriented architectures are represented in previous chapters. However, it seems useful to mention some geospatial specifications and protocols for particularly supporting the design of 3O-HIS which are needed to be considered when designing a service-oriented architecture such as Web Services Description Language (WSDL); Universal Description, Discovery, and Integration (UDDI) and SOAP. WSDL forms the basis for the original Web services specification. In some cases, Universal Description, Discovery, and Integration (UDDI) used

to express a registry of services. Zaslavsky et al. (2007) represents one of the key requirements for a hydrologic information system is the ability to easily discover, retrieve, and interpret hydrologic time series for solving the interoperability issues between the datasets with other disparate datasets. Integrating hydrologic data across agency boundaries and combining them with an observation time series collected by multiple academic research projects has been challenging due to the extreme heterogeneity in how the different repositories are organized, described, accessed, and maintained.

3.6 Ontology in 3O-HIS

In 3O-HIS, ontologies is used for semantic resources registration and transmission between services. Ontology likes a material transport in services. The mechanism of ontology transport in systems is important and it supports the processing of ontologies transport validly and effectively in the hydraulic

systems. During the processing ontology transportation, the criteria or protocol has been developed to identify the ontology for validity, classify and retrieval. It is necessary to develop concise protocols for standardization services for data interoperability. The criteria need to be in common use that the services which can meet a common criterion. Users on client face conveniently in finding suitable data to solve the semantic heterogeneities.

Ontology is essence in 3O-HIS and ontological differences between scientific disciplines are challenges to represent one science to others and to process the view of the nature of the world comprehensively (Argent, 2004). As mentioned before, HIS centre is a vital services in CUAHSI HIS. Using of ontologies that is a key in the development of HIS centre can partly solve this issue and over 96% of repositories are available for ontology-based discovery and they focus on the ongoing

ontology development and want to establish community consensus mechanisms (Zaslavsky et al., 2007). However, the existence of huge information repositories is inconsistent created by different specifications and protocols, and prevents information flow to go further between services. For example, Zaslavsky et al. (2007) stated that the existence of huge information repositories created by traditional applications prevents information flow between the external and internal parts. After summarizing environmental modelling issues both technically and conceptually, the inconsistent issues have been one of the main bottlenecks to design modelling frameworks and component-based modelling. It is necessary that hydraulic data and modelling as services have free connection that service interface is independent implementation.

Using ontologies for pairing these traditional repositories in HIS centre is challenging that replaces these huge sources by loose

coupling services and satisfies the required information flow using SOA (Zaslavsky et al., 2007). In time consuming side, although using SOA in the long term is time consuming, it leads to save on costs, while it causes high costs in short term usage. For example, other extra costs are consumed including involved as a result of promoting the web and security technologies, training in, learning new system, and managing organizational system(Verma et al., 2012). In data maintenance side, due to open communications which exist besides web services in sharing information among the staff, customers, partners and others, the security of an organization is compromised and in need of strong security mechanisms. Zaslavsky et al. (2007) stated the challenge of maintaining the central metadata catalog in synchronization with multiple sources of hydrologic data. The challenge of maintaining the central metadata catalog and lack of information exchange standards may have the issues of time consumption, potentially when connecting one from huge

repositories and maintains ontology for synchronization with hydrologic data in repositories. These standards and structural framework necessitated towards better alignment and interaction.

3.7 Summary

This chapter presents a 3O-HIS platform and the methodology of encapsulating hydrodynamic modelling (e.g. Telemac) in 3O-HIS. A new service-oriented web application 3O-HIS which is derived from CUAHSI HIS has been developed that services interaction by full use of existing and emerging standards to aggregate heterogeneous geospatial data and observational data, and deliver rich user interface for users. This innovative way performs analysis tasks at service-side and represents results at client-side.

The vital part in CUAHSI HIS called ‘HIS centre’ has been

removed by fully use of SOAP, WSDL and XML standards to enable users and applications to search across a comprehensive catalog of hydrologic time series available from data repositories. Integration of hydrodynamic modelling with 3O-HIS, we need to understand the requirements in both technically and conceptually. In technical aspect, the key limitation is that models are varied and consists of different inherent codes and algorithms which are hidden. In code and preference level, inherent TELEMAC framework hardly transplant into new 3O-HIS.

The architectures of 3O-HIS have three layers: Data layer, Service layer and protocols, compare with CUAHSI HIS, ‘HIS centre’ has been removed, and the structure of 3O-HIS is more concise. In particular, the structure works well for data-intensive hydrologic research, such as massively hydrologic data varied with time series. The system configuration and definition of key

input files are represented which relates the service bus. The development of 3O-HIS provides a flexible, rapid and open way and service interface that allows differently modelled parts of the system to work together. Finally data repositories, SOA and data repositories with SOA are represented including the principle of definition geometry file and hydrodynamic models which use to define input files preparing the Blackpool and Demra case studies.

Chapter 4 Data management

4.1 Introduction

This chapter defines a major link to resource with SOA and highlights the challenge during data transmission between services. A large datasets of model results have been generated in order to understand the circumstances processing under various conditions. The accumulated datasets need to build a comprehensive datasets manage frame that has a capability to access, organize, integrate, visualize and analyse these data. Several large geosciences organizations have made attempts on the implementation of searching and integrating information across different data repositories such as NEON, LTER and GEON. 3O-HIS likes other data repositories, has the capabilities of data discovery, data retrieval, and data visualization. They are

the basis of new comprehensive infrastructures known as “cyberinfrastructure” that allow the research team to share distributed data resources through high-speed networks. Cyberinfrastructure has been developed by many activities and organizations to support the needs of multidisciplinary collaborative research, such as the National Ecological Observatory Network (NEON) (Keller et al., 2008), the Long Term Ecological Research Network (LTER) (Bain et al., 2012), the Geosciences Network (GEON) (Richard et al., 2011), Geospatial One-Stop (GOS) (Choi and Yu, 2008), and Consortium for the Advancement of Hydrologic Sciences Inc. (CUAHSI)’s Hydrologic Information System (HIS) (Maidment, 2008). And our 3O-HIS is developed based on CUAHSI HIS as a prototype.

All these programs are under a same cyberinfrastructure concept; meanwhile each of the organization has its own focus. NEON

usually cover U.S. ecological variability, LTER aimed at understanding processes in a wide range of ecosystems in a long-term. Geospatial One-Stop is used as geoportal to find geographic information; the prototype was the basis for the OpenGIS Portal Reference Architecture. Geoportal server used by local government consists of data of various formats. GEON is a research program for the earth science community. GEON Portal, a web-based distributed resource management system, provides integrated access to data and tools needed for knowledge discovery in the Geosciences.

It is important to understand the workflow in these database systems. Each processing between services can be treated as message transmission between client/server. For example the assumption of the workflow starts from a user who wants to request data resource such as a map. User access at client side, request information sends to server, server may send the map

client side or send the request information to other services depend on the data location. In addition, users also can upload the map to a service from the client side.

4.2 Major linking resource with SOA in environmental science

SOA assists interoperation among various information resources. Different types of resources have emerged in many disciplines related environmental science, including in the solid earth sciences (The Geosciences Network, or GEON: www.geongrid.org), ecology (The National Ecological Observatory Network, or NEON: www.neoninc. org), oceanography (The Ocean Observatories Initiative, or OOI: http://oceanobservatories.org), and atmospheric sciences (Linked Environments for Atmospheric Discovery, or LEAD: http://lead.ou.edu). Environmental databases are varies and each one has own focus. For example, NEON and LTER are the

database more related with ecological ecosystems, GEON is the database for the earth science. For the requirements of integration database, several systems has been developed recently such as Geospatial One-Stop (GOS) and CUAHSI HIS.

GOS is a web-based portal that provides public access to geospatial information as part of a United States e-government initiative. GOS aims to promote coordination and alignment of geospatial data collection and maintenance among all levels of government. For the purpose of GOS, GOS seems most appropriate for design 3O-HIS based on web services. However, semantic heterogeneities may occur, because all levels of government uploaded data do not follow a standard, the criteria are not met for all government and user on conventional keyword-based search techniques that data is same and redundant. For example according to the 2001 initial business case for GOS, about 50 percent of the federal government

maintain the same or similar information because of each federal government established own standard for data collecting.

In CUAHSI HIS, HIS Central is physically composed of numerous servers and supports users to request data in ‘hydrodesktop’ clients and to get access to data through data discovery. Resource are used to define in a HIS Central service. Source schema provided by HIS Central can instance of ontologies commonly and the resources associated ontology concepts, spatial and temporal extents can be found according to a user’s query.

4.3 SOA with database

A key advantage by using SOA is avoiding code duplication (Papazoglou and Georgakopoulos, 2003). It depends on the protocols defined in services to direct the target information precisely. Each subservices are be interoperate with consistent

protocols under different system operation environment (e.g. windows, Linux). A loose-coupled structure of application is flexible for services adding or removing. New services can be easily created and then consumed within client applications. All communication between services and clients is through a defined interface, there is platform independence between clients and servers. Loosely-coupled services are possible for developers to reuse the same underlying services within multiple applications (Huhns and Singh, 2005) and the application can process across multiple environmental systems. For example, UK real-time sea level data has been obtained from National Oceanography Centre, the sea level data which is investigated by observation station inputs into Telemac model to predict the sea levels at each node in the mesh of Blackpool. Compare with the Linux operating system which supports Telemac processing, the results of sea levels at each node can have an analysis for policy makers, planners and coastal engineers under ArcGIS

10.0 in Windows operating system. Using FORTRAN on a Linux operating system for TELEMAC calculation and service could still be consumed by a Windows client

The detail code of application is hidden and the standard messages passed between different modelling system and the service. The service execution remains external to the modelling system; therefore it maintains maximum integrity of modelling. SO modelling allows for a hierarchical representation of complex water resource systems. A service can be a conglomeration of other services, allowing for different levels of system abstraction. In some cases modelling water resource systems requires a coarse view of the natural world (e.g. decomposing a system into atmosphere, land surface, subsurface, and other model services), while other cases may require a more detailed view into individual process-level components (e.g. infiltration, evaporation, overland flow, and other process-level

services). Coarse decompositions minimize the complexity in setting up a system representation as a workflow by specifying how data is exchanged between system components. Because tight-coupling approaches are generally more computationally efficient than loose-coupling approaches, coarse decompositions also improve computational performance (Pingali and Stodghill, 2004). Refined decompositions allow for increased flexibility because smaller units within the modelling system can be more easily added or removed. Ideally, systems would be represented by a hierarchical structure where coarse services are themselves workflows consisting of more refined services. For example, ideally TELEMAC may define as refined decompositions in a hierarchical structure of 3O-HIS. We have not achieved linking TELEMAC with refined services because of limited understanding of TELEMAC hierarchy in a distributed manner and the system difference increased the difficulty of combining.

We can also extend the theory that refined services, processing are synchronized which implicates the use of high performance computing (HPC) through computational architectures like the Grid (Foster et al., 2001). The Grid provides important extensions for service-oriented modelling including state and fault handling, authentication, and resource management. Even without Grid computing, services can aid in improving model performance and efficiency. For example, web services could be used to expose a model with large data input requirements but relatively small output datasets. A spatially distributed watershed model, for example, might be exposed as a web service, keeping the terrain, soil, and other parameterizations of the model on the server side and only transmitting soil moisture outputs for client applications. In such a case, the model would be stored geographically near the large input datasets, saving the end user that is only interested in soil moisture conditions from having to download and process these input data required for

running the model locally.

4.4 Challenge for design SOA with data repositories

To review the related works mentioned above, several challenges can be summarized. From the system design point of view, the primary challenges of a web services approach to environmental modelling are related to the loosely-coupled, web based architecture of services that can result in performance, reliability, and security issues. Performance challenges in using web services for water resource modelling, primarily relate to tightly coupled process interactions that may require large data transfers, particularly when initializing and parameterizing the modelling domain, or from computation tasks with long computation times. In addition to performance issues, reliability of services can also be a disadvantage of service-oriented modelling approaches. There is the possibility of remote servers becoming temporarily unavailable, thus disconnecting all client

applications dependent on that service. Finally, security must be considered to prohibit unauthorized users and track overuse and abuse of services. Many of these issues are addressed by existing technologies such as Grid infrastructures (Foster et al., 2001) that can be used to enhance the applicability of service-oriented architectures for modelling. In many, but certainly not all modelling scenarios, these disadvantages of service-oriented approaches can be minimized through thoughtful system design (Pingali and Stodghill, 2004). When designing services, it is important to consider the response time and size of messages passed over the Internet. Model processes that require numerous communications with data transfers during runtime should be tightly-coupled within a single service to avoid network latency. Intelligent caching of data can also be used to minimize data transfers of repetitive information. For example, if a large dataset is required to initialize a modelling domain within a service, the data can be maintained within the

service’s state so that it does not need to be passed to the service with each service method call.

In application point of view, the main barrier to make full use of data resource on services is that users are unable to rapidly access data sources and retrieve appropriate data sets from different data repositories. The issues can be represented as syntactic and semantic heterogeneity among data from different sources (Goodall et al., 2008). Syntactic heterogeneity is resolved by converting proprietary data formats into an XML document. Schematic heterogeneity is faced by adopting the OGC standard service protocols, such as OGC Reference Model (ORM) as well as Catalogue Service (CAT). The OGC Reference Model (ORM) describes a standards baseline consisting of abstract and implementation standards. CAT supports the publication and searching of collections of descriptive information (metadata) about geospatial data,

services, and related resources. It needs attempt to extend the standards for OGC's services including the WMS, the WFS, and the WCS widely implemented to facilitate interoperability. Because the semantic heterogeneity issues has been addressed in recent years. Ontologies are extensively used which can be seen as best solution and are clearly defined by protocols for improving the search capability, alleviating the semantic heterogeneity issues between repositories and enhancing interoperability between them (Beran and Piasecki, 2009). To obtain an actual ontology becomes another significant challenge with the increasing growth in popularity of web services. Ontologies needs be viewed from locating web services by enabling robust queries. A key important thing with the above technologies is purely syntactic that finding an actual ontology relies on software developers to understand the intended meaning of the descriptions and to carry out the activities related to web service usage; thereby OGC's services do not include

automatic service discovery and invocation and fully machine-understandable semantic descriptions of web services are necessary to provide support for on-the-fly discovery.

4.5 Data conversion

The data conversion is not simple and takes lots of efforts, and is a core part in 3O-HIS. Based mainly on data conversion, TELEMAC model is encapsulated in 3O-HIS seamlessly. 3O-HIS linking Telemac as an example, three types of mandatory files: a **Steering** file, a **Boundary** file and a Geometry file are required to convert appropriate format in 3O-HIS. The **Steering** file and **Boundary** file are ASCII text files that each line has no spatial meaning, and researchers hardly set up the relations between map and text files. In 3O-HIS, the value of parameters must have the geographic reference system in order to associate with mesh node locations. The lack of the spatial capacity in TELEMAC input files

implicates the requirement of conversion from a Steering file or/and a Boundary file to shapefiles.

4.5.1 Steering file conversion

Conversion from Steering file is straightforward. Steering file is Text format and, the text has been divided into two groups by different conversion ways. The first group includes STEERING FILE ='MerseyDeeopen.cas'; BOUNDARY CONDITIONS FILE ='Lbay2.bc'; GEOMETRY FILE ='Lbay2.geo' and RESULTS FILE='LbayTel.res'. In the first group, data are variable for different cases. For example, GEOMETRY FILE parameter often changed for Telemac collection. The other group represents a constant type such as MASS-BALANCE =YES, we defined MASS-BALANCE parameter as a standard that all services should comply with. The value usually is constant and stored in PostSQL database. It is important to notice that the environment is totally changing from MS-DOS to

distributed services by Java IDE.

In 3O-HIS, some parameters in Steering file may not be suitable for sharing to all users. For example, FRICTION COEFFICIENT is not appropriate to share with others because different location has different type of soil and frictional resistance is various. FRICTION COEFFICIENT parameters defined as variable type.

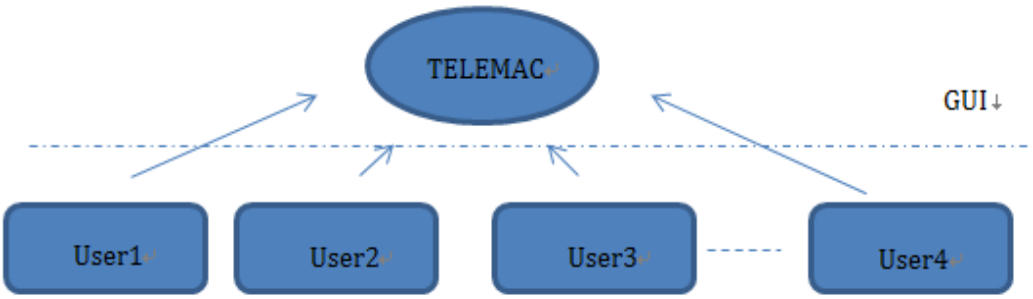
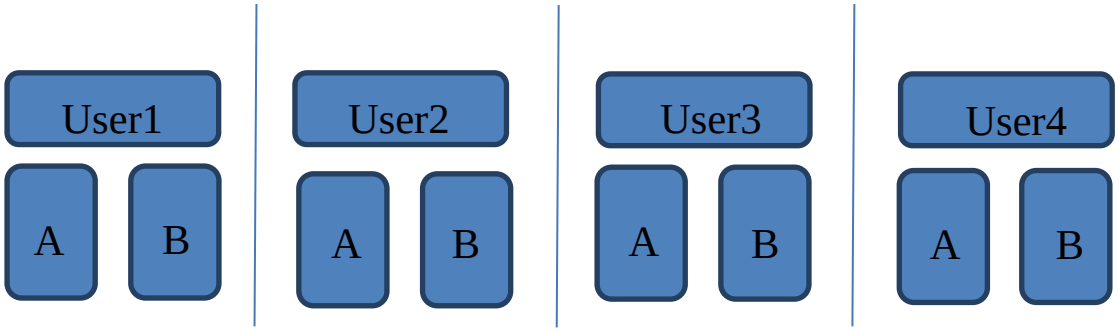


Figure 4.1 Distributed flowchart between users and TELEMAC

Following the instructions shown in Figure 4.1, the GUI can build a uniform ‘Steering’ file for each service. Usually GUI is used to set different variable physical parameters for different users as a uniform format under 3O-HIS. For example, it is necessary to set the difference of time step for each user in a

national scale analysis. The mesh is divided into several pieces; some areas with a fine mesh that researchers want to understand the environmental processing clearly such as Liverpool Bay and River Wyre in Fleetwood. And some areas may have a rough mesh; the time step may be changed in different meshes based on the complexity of topography. Therefore, it is necessary to develop a GUI interface for Steering file and Boundary file conversion.



(A): common standards or constant value for all users

(B): specific value in certain service

Figure 4.2 Type of value stored in services in 3O-HIS

The function is related to the type of value stored in each service (see Figure 4.2). Common standards in (A) will be stored in SQL service and other services invoke these common standards

directly under a same project. Usually the information is stored and shared in one services. It defines some specific values individually in (B), which cannot transmit to other users to ensure the services receive valid data. Because if we share the Geometry file from user1 to user2, the inconsistency between geometry file and others, such as boundary files or FORTRAN file may lead the interruption when TELEMAC model is initiated. Balancing the values stored in public or private can reduce the amount of work for data collection and facilitates using the TELEMAC model in an integrated framework for a large scale.

4.5.2 Boundary file conversion

Boundary file conversion is complex because there is no spatial information in text (.txt) format. Firstly, the positions of each point on the mesh need to be obtained. Value of position is saved in the database. PostgreSQL and PostGIS add-on is selected to

be the database management system (DBMS) as the basis of the mesh database. PostgreSQL can store the geographic objects spatially and has the capability of performing complex queries and supporting various programming languages (e.g. Java).

4.5.3 Result file conversion

The most commonly used format is the Selafin format for results stored. The internal standard format described in TELEMAC2D user manual. In this section, it focuses on the conversion from Selafin format to PostSQL spatial database. A solution for conversion from res to PostSQL is to use interfaces shown in Figure 4.3. Res file placed in the middle is the Selafin format from Telemac model. The entrance of the 3O-HIS is the main function in lib.read_fortran_files; lib.OpenFile files used to check the validity of the res file. It defines the operations' signatures, and check the size of the res file. The lib.read_file_code is the main part of res conversion. It is

indicated that a number of get functions are created for each physical variable which is linked to the item defined in res file. Most of them are mesh information such as getNELEM() is the number of elements in mesh; getNPOIN() is the number of point in mesh; getX() is the X coordinate of each point and getY() is the Y coordinate of each point.

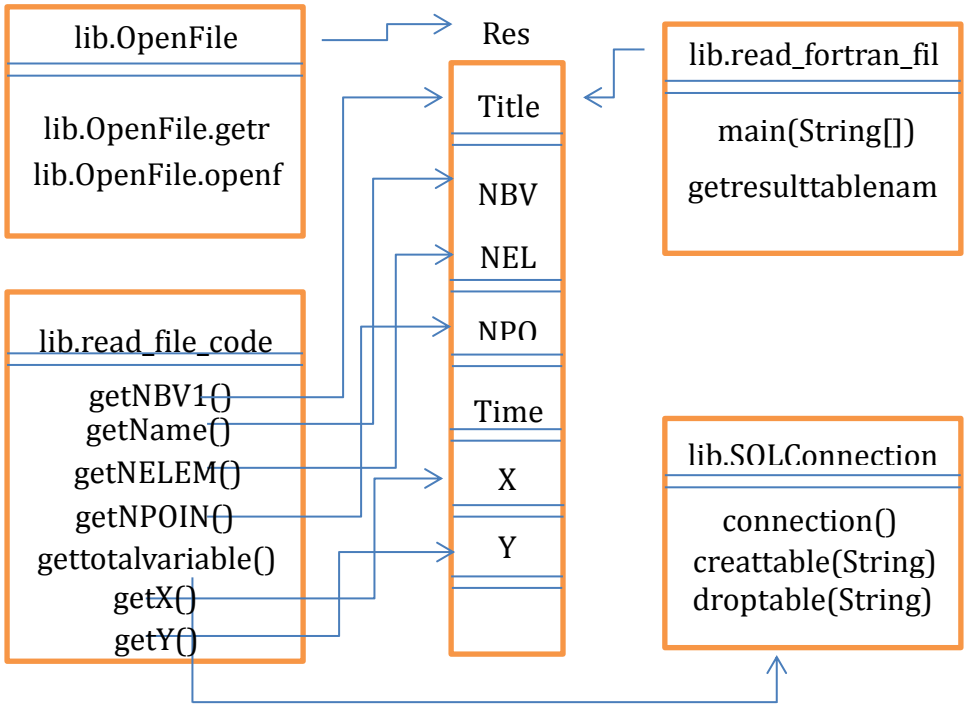


Figure 4.3 The mechanism of tranmision from res to PostSQL

Figure 4.3 shows an example which utilizes a collection of tables to represent the result of the TELEMAC and the corresponding conversion to 3O-HIS. The TELEMAC model

code solves depth-averaged free surface flow equation from Saint-Venant. The result can be predicted based on the last time step. Therefore, all points obtain the value of all physical parameters and then prepare a calculation from the boundary for the next moment. The calculation is a recursive loop. Figure 4.4 represents the structure of the result after a finite number of iteration. It shows the result at time1, time2 and timeN which is physical parameters for all points after time step×n. It is a challenge to transfer the structure of data storage into 3O-HIS. Generally, the study aims to understand the tides processing in specific areas. It means we make an effort to focus on physical parameters such as variable in the time series in specific points. It consumes a massive memory. For example, the water depth at point 2 is a specific location. We need to link all tables (time1-timeN) and fetch the value (it is in Row 2 and column H). Because all tables cannot close until the timeN table is end of loading, the system will collapse due to out of Java Heap

Memory.

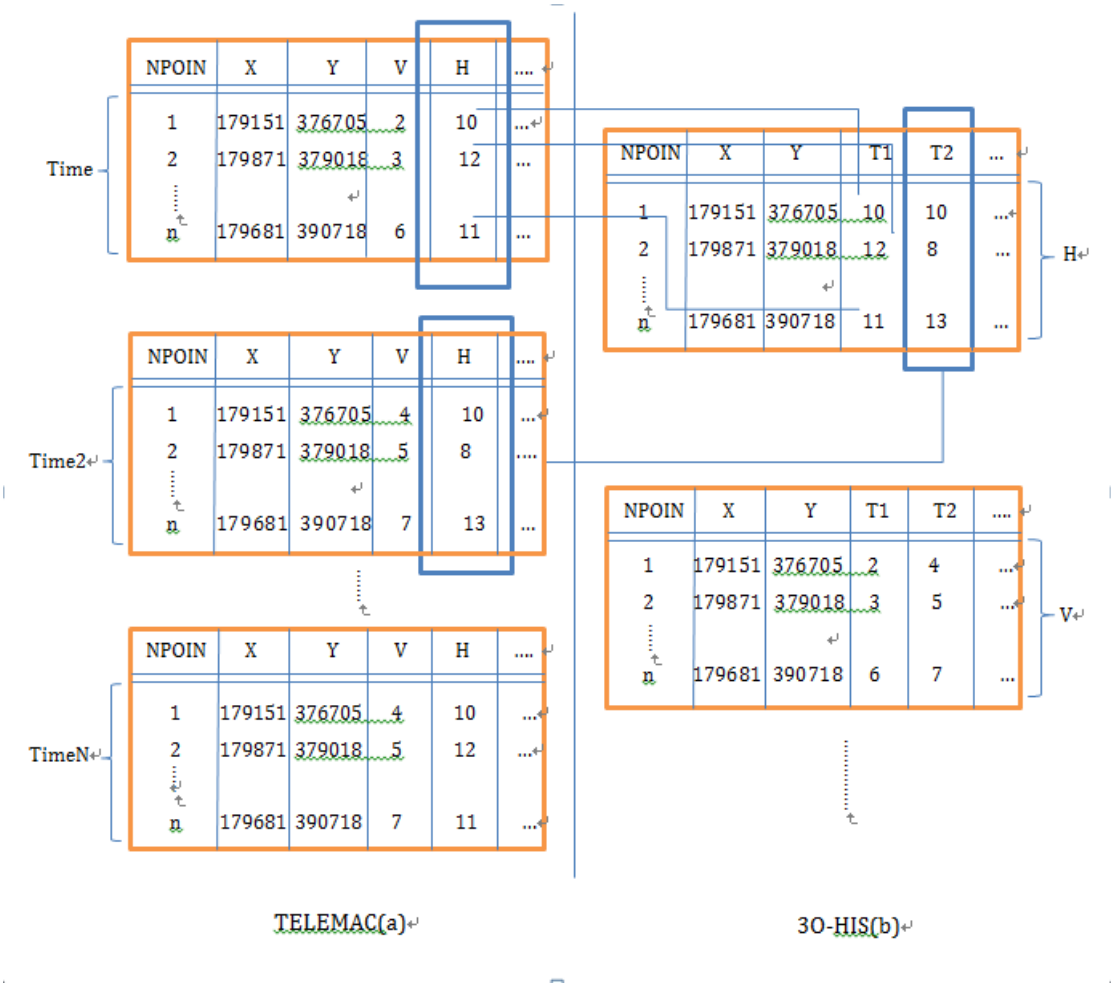


Figure 4.4 Result data conversion from TELEMAC to 3O-HIS

Physical parameters are represented in the time series. A 'container' is built when data conversions begin. The 'container' operates gettotalvariable() function to obtain an array named totalvariable. The array of totalvariable is important as it includes the value of the each physical parameter. The structure

of array of totalvariable in container is shown in Figure 4.5. The array of totalvariable is a three-dimensional array: name of variable, number of points and the Number of time steps. NBV1 represents the name of the variable, and it determines a specific physical parameter selected for conversion. NPOIN is the value of points, and time steps are the maximum size, number of time steps for conversion. We suggested the size is no more than 200, because after 200 iterations, the array of totalvariable is bloated and time consuming. The default of iterations is 150, after 150 times, the value in the array of totalvariable will link the lib.SQLConnection and write into PostSQL. After this, the container is clean for next 150 iterations. This design in newly 3O-HIS is suitable for data storage and avoid out of Java Heap Memory.

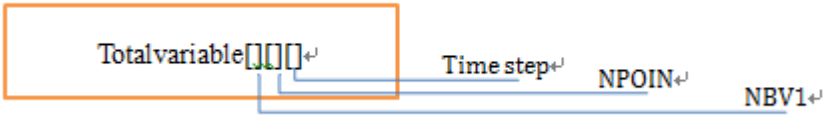


Figure 4.5 Structure of Totalvariable

4.5.4 Data transmission

Data transmission uses a uniform standard: XML that each distributed services can store geospatial data, semantic descriptions and geoprocessing service chains based on it. Based on the instructions, users can distribute meshes in different computers through web servers under the framework of 3O-HIS.

4.6 Key input files stored in service

4.6.1 Geometry file

Topographical information is a critical factor and significantly influence the patterns of floodplain inundation (Marks and Bates, 2000), meaning that high-resolution elevation data with improved spatial accuracy can provide significant advantages in predicting inundation patterns. Usually Light Detection and Ranging (LiDAR) and Interferometric Synthetic Aperture Radar (IfSAR) are two notable remote sensing technologies to provide an accurate topographic map sources (Jorgenson and Brown,

2005). Topographic maps are provided by Marine Digimap (EDINA), The Bathymetry & Elevation (BE) have been used which describe the natural shape of the Earth's surface of land and under the sea and includes height contours and digital elevation models. The coastal line gaps existing in the DEM, in the sea levelling system the reference of the depth data approximates to LAT, for land survey datum the reference of depth data related Ordnance Datum (Newlyn) which is the point of origin corresponds to the average value of Mean Sea Level (MSL) at Newlyn during the years 1915 to 1921 (EDINA) were used. The difference between two data sources were adjusted by manual editing.

4.6.2 Hydrodynamic models

Previously, local assessments of coastal flood hazard were based upon detailed 2D or 3D hydrodynamic models that simulate hydraulic changes in water flow and level resulting from

specific benchmark events (Battjes and Gerritsen, 2002). The accuracy of the result depends on the detailed hydrodynamic models and conditions such as boundary conditions and the interpretation of hydraulic roughness and bottom topography from bathymetric surveying. Potentially, an accurate result requires a detailed description of initial boundary conditions and computational mesh. It is computational and complex that models iterate over successive time-steps using differential flow equations across the mesh. In hydrodynamic models with GIS fields, simpler finite-difference (raster) models have been developed and Bates and de Roo (2000), Horritt and Bates (2001) stated that raster models were quicker than finite-element models.

In raster models, a grid mesh of 10^6 cells has been developed in Horritt and Bates (2001) studies, the elevation of each cell is crucial that it determines the water flow by comparing with the

elevation of around 8 cells. The simple raster model takes advantage of Geographical Information Systems (GIS) which can provide an effective framework to integrate and analyse disparate environmental data sources, especially more general methods of strategic flood-risk assessment can be processed over larger areas. However the accuracy of inundation map produced by simple raster model is not as good as numeric 2D or 3D hydrodynamic models. Thus, Thumerer et al. (2000) developed another way that identifies of distinctive flood characteristics within a vector GIS. This approach can easily identify some hydraulic characteristics such as velocity, depth, features represented by points. However points, lines and polygons are difficult to represent spatial patterns in flooding.

Developing 3O-HIS would take the both advantages, considerable hydrodynamic models can be encapsulated and grid-based (raster) GIS can distinguish spatial variation and

spatial patterns in flooding with an appropriate resolution especially for larger areas for assessing flood risk. This chapter will explain in detail how to deploy the 3O-HIS by understanding the case study of flood risk assessment in Blackpool.

In Hydrodynamic models, the Telemac model gives insight into the development of the hydrodynamic processing such as flooding over land leads an inundation depth. The model consists of a two-dimensional flow model that is based on the Saint Venant equations. So WPS technique has been used that Telemac model is encapsulated to operate on spatially referenced data. It specifies three operation GetCapabilities, DescribeProcess and Execute and make applications easier to collect a lot of input data that represent the natural shape of the surface of land and under the sea.

4.7 Summary

This chapter represents data repositories, SOA and data repositories with SOA in many. Details features of data repositories and SOA structures have been represented such as client/server; coarse/refine. Any processing between services or between user and services can treat as client/server. The client sends the required information to the services from the portal which is defined by data repositories, and services may link other services or send the result to the client. Currently, the protocols between client and service are XML, GML, and SOAP in 3O-HIS which explained in other chapters. Coarse/Refine is an important concept of design SOA with data repositories. The modelling built in services normally is no single framework, because of the computational algorithm and large number of parameters. The refine decomposition is needed. On the other hand, Coarse/Refine concept may support the HPC and Gird computing, because the small refine processing are

synchronized.

This chapter also represents the challenges of using loose-coupled SOA for model/data communication between data repositories or between applications. Reduce the duplication and problems for adding or removing services, conglomeration of other services with Coarse/Refine, and processing in different environmental system. An application of hydrodynamic model enclosed in 3O-HIS is discussed which two case studies are represented in chapter 5.

Chapter 5 Application

5.1 Introduction

This chapter presents the applications of GIS in water resources and hydrological modelling. Two Case studies: Blackpool in England and Demra in Bangladesh are presented to implicate the benefits that integrate GIS and hydrodynamic modelling in 3O-HIS platform. Assessment between GIS and hydrodynamic results under web services indicate a promising future for GIS application in hydrological modelling.

Section 5.2 discussed the current states of integrating GIS with hydrological modelling, and general user guidance for deployment of services in 3O-HIS are explained in section 5.3; other different existing hydrological modelling to deploy in

3O-HIS are discussed in section 5.4. Section 5.5 shows case studies such as Blackpool in England, and Demra in Bangladesh for a better understanding of the advantages of integrating GIS and hydrological models in section 5.6. Section 5.7 comparing the other open standard GIS with commercial GIS, it represents the interoperability of 3O-HIS for large distributed data analysis and modelling. Finally a summary reviewed in section 5.8.

5.2 Current states of GIS with Hydrological Modelling

GIS has evolved dramatically and emerges to combine with statistical analysis, and database technology due to its powerful ability of spatial data analysis (Foote and Lynch, 1995). It is convenient to capture, store, manipulate, analyse, manage, and ultimately present all types of geographic spatial data. The ability of spatial data analysis feature in GIS can help researchers analyse and/or present huge amounts of data, and GIS can have different applications and hence is of high interest

in various fields of study. Water Resource Engineering indeed needs to present modelling and analysing spatially distributed data with different spatial resolutions (spatially and temporally distributed). Therefore, GIS is indeed a suitable tool for solving water resources problems.

Liang and Molkenthin (2001) firstly take advantage of the evolved network communications, developed a virtual GIS-based hydrodynamic model in a distributed environment. After that Argent (2004) also discussed the great potential for distributed computing and databases, as well as distributed communication achieved between the clients and server. However, there are several limitations in existing clients-server GIS-based hydrodynamic model. New challenges include data collection infrastructure at unprecedented scale, management of increasing volumes of data as the temporal and spatial frequency of data collection increases (Horsburgh et al., 2011). The major

issue was large datasets exchanging inefficiently between services in WWW platform. The issue becomes more complex when the hydrodynamic model and the mesh need a very fine resolution for consideration of different scenarios such as various boundary conditions, sea level rise and economy-societal factors (e.g. land use, cost of buildings).

Limited studies have discussed the solutions. The previous studies have demonstrated the applicability of utilizing GIS as an integrated environment for manipulating hydrologic. The fact makes a difficult situation that observatories are geographically distributed and run by disparate research groups. With the increasing volumes of data, it is challenge that sharing environmental data in Internet across observatories. It is currently complicated that data can be published for cross-observatory analyses. To overcome the obstacles, a new 3O-HIS has been developed which considered as service

orientation linked to GIS. The desire of creating 3O-HIS is establishing an interactive system that heterogeneity of data can be discovered from more than one observatory and hydraulic model as an encapsulated component.

5.3 User guide of 3O-HIS

In this section, a general user guide has been written for discussing how to use 3O-HIS. Lot of efforts has been made in deployment of services in 3O-HIS, to ensure other researchers can follow the deployment for creating their own hydraulic model-GIS system, there is a requirement to represent a guide including how to deployment services in 3O-HIS. The features of data in 3O-HIS can be interoperation, communication, and publication ultimately present Telemac modelling encapsulated as services.

A series of software has been listed in section 5.3.1 for

preparation for input. And how to deploy the services is discussing in section 5.3.2. Finally GUI of 3O-HIS discussed in section 5.3.3.

5.3.1 Installation

For successfully managing, analysing, and publishing results of numerical models, particularly in cases when the datasets are extremely large, software which needs to install to set up the environment for services deployment is listed below.

- Tomcat (test version 6.0.35)
- WPS-52n
- Notepad++(http://notepad-plus-plus.org/)
- Geoserver (Optional)
- Eclipse (Optional)

The five listed software need to install properly for developing 3O-HIS. The Tomcat and WPS-52n are obligatory which used setting up the system environment. Tomcat is an open source

software implementation of the Java Servlet and JavaServer Pages (http://tomcat.apache.org/) for the client/service environment. WPS-52n is one of WPS applications for services processing (http://52north.org). Notepad++ is a free code editor which we use it for editing XML, GML and js[5] files. Geoserver is a web service with OpenGIS specifications. Eclipse is a platform for Java and plug-in development plus adding new plugins. Before we test many brewers as client, Foxfire browser is the best one that has less client response times and the feature of ‘fire bug’ can have a debugging during the design step.

The mechanism of processing flow with software is shown in Figure 5.1. An assumption of software installed successfully which bundles Tomcat (6.0.32) and WPS-52n in services. Tomcat is used to deploy WPS online and WPS-52n provides

5 Js: js means **Java Servlet**, in this dissertation it created particularly for processing Openlayer software in Foxfire browser which is an open source software for displaying map in browsers.

the method to processing geographic features. A processing flow has been charted for understanding the relationship between software, as shown in shown in Figure 5.1

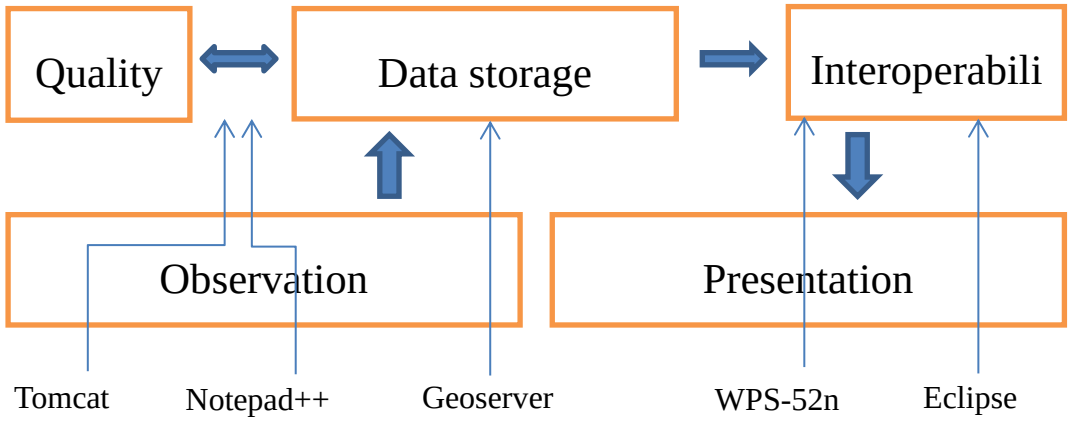


Figure 5.1 The processing flow of 3O-HIS

From the Figure 5.1, each software has own function in specific place in the chart. The orange colour box represents the general architectural and procedural components which include the following: 1) data observation infrastructure which collects observations by the sensors and telemetry systems. 2) Data storage represents data systems and software required for creating a persistent repository. 3) quality assurance is the processing that transitioning from raw data to publishable data

products 4) interoperability: this is the key processing in the flow that data are processing following WPS technology and publishing the data in interoperable formats and 5) presentation: the tools data consumers use to get the data for the purpose of creating visualizations and analyses. Tomcat is used to deploy WPS online and notepad++ edits the XML and GML files for the messages transmission, they are shown as blue arrows. Geoserver is one of services used to storage dataset. One reason for choosing Geoserver is open source server. It desires for interoperability, and supports any major spatial data source using open standards. Here the version of Geoserver needs to be consistency with the version of ‘PostgreSQL’ which is used for storing spatial datasets. Any version of ‘org.postgresql.Driver’ inconsistency may lead the connection to fail. WPS-52n provides the method to processing geographic features. The Openlayer package (http://openlayers.org/) can be used as representation tools for publishing the data in interoperable

formats. Using Openlayer code for deploying hydrodynamic parameters on services. There is an example for deploying Blackpool vector map on Geoserver and representing on Openlayer, it shows in the line:

```
var Blackpool _2002_shp = new OpenLayers.Layer.Vector("Blackpool _2002_shp ", {
        projection: new OpenLayers.Projection("EPSG:4326"),
        strategies: [new OpenLayers.Strategy.BBOX()],
        protocol: new OpenLayers.Protocol.WFS({
        url:   "http://localhost:8081/geoserver/WPS_Water_Discharge/wfs",
        featureType: " Blackpool _shp",
        featureNS: "http://www.opengeospatial.net/WPS_Water_Discharge",
        version: "1.0.0",formatOptions: {outputFormat: 'GML3'},outputFormat: "GML3",
        readFormat: new OpenLayers.Format.GML.v3(gmlOptionsIn)})
        });
```

For successfully processing the version of Openlayer need to be defined that index.html file in the folder (apache-tomcat-6.0.35\webapps) needs to modify and change the line:

```
<script type="text/javascript" src="http://openlayers.org/api/OpenLayers.js"></script>
```

to

```
<script type="text/javascript" src="http://openlayers.org/api/2.9.1/OpenLayers.js"></script>
```

Eclipse provides the platform for compiling hydraulic models with Java code. The services include hydraulic models that can

have a numerical processing between services. In Eclipse, the polygon output (Geometries) are created using GML2 standard. In GML2 standard, the tag pair of the_geom use to store the polygon geometry as shown in Figure 5.2.

```
<wps:the_geom>
   <gml:MultiSurface srsDimension="2" srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
      <gml:surfaceMember>
         <gml:Polygon>
            <gml:exterior>
               <gml:LinearRing>
                  <gml:posList>90.51747205200007 23.73928531100006 90.51684153400004 23.73924263200007
                  90.51654482000004 23.739251168000067 90.51622028900005 23.739276775000064 90.51593160700003
                  23.739261034000037 90.51624888400005 23.73951151600005 90.51639165100005 23.739701873000058
                  90.51677236500007 23.74001120300005 90.51717687300004 23.740320532000055 90.51750999800004
                  23.740582273000086 90.51793830100007 23.741319906000058 90.51793830100007 23.74127231700004
                  90.51796209500003 23.741367495000077 90.51839244000007 23.741419137000037 90.51839001200005
                  23.74097536700009 90.51800057400004 23.74095829500004 90.51800984700003 23.74059126500009
                  90.51761113700007 23.740548587000035 90.51752768600005 23.740258376000043 90.51749059700006
                  23.73995109300006 90.51744423500003 23.739712095000073 90.51749059700006 23.739362132000053
                  90.51747205200007 23.73928531100006</gml:posList>
               </gml:LinearRing>
            </gml:exterior>
         </gml:Polygon>
      </gml:surfaceMember>
   </gml:MultiSurface>
</wps:the_geom>
```

Figure 5.2 An example of geom feature representation in GML2

The tag pair of gml:posList store a number of values which represent the location of one polygon vertex . All value together in the gml:postList represents an unique polygon to the related geospatial mesh. Nine maps were uploaded to Geoserver from 2002 to 2010, and they generate a GML2 format to describe

geographic features. After the Geometry stored in services, the topological information can be retrieved from any location or remote computer under the framework of 3O-HIS. In Figure 5.3, polygon features are represented as the agricultural areas in 2002.

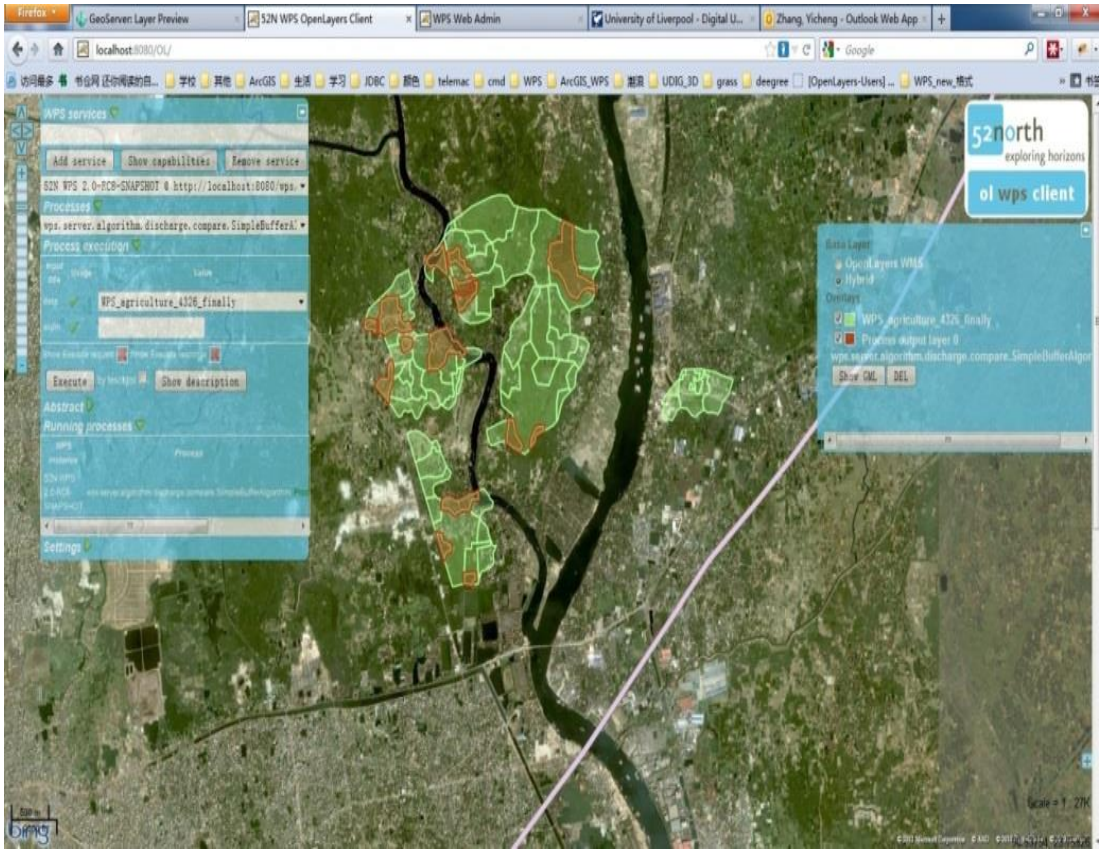


Figure 5.3 Display of Catchment area using 52 °North WPS

5.3.2 Using the 3O-HIS

The graphic user interface (GUI) (Figure 5.4) is created for

linkage the results and geographical features. Setting appropriate parameters for running hydraulic models. The Shapefile is composed of geospatial vector data in 3O-HIS. Mesh files and boundary covers shapefiles and some physical parameters such as boundary files stored in PostSQL. A Shapefile usually stores geospatial vector data format.

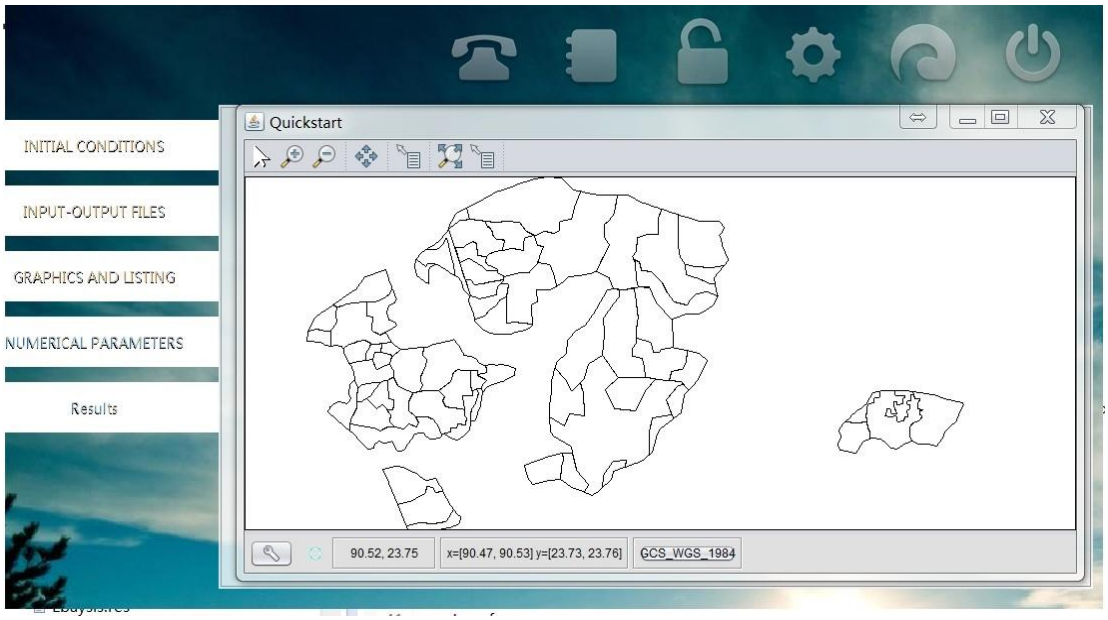


Figure 5.4 GUI for TELEMAC

The linkage of PostGIS with Shapefile as shown in Figure 5.5. Connection component was developed for saving model simulation results into the PostSQL database. The connecting method has been developed using Java that 3O-HIS provides a

feasible functionality for hydrodynamic results connecting with local or/and remote services. The development goal for this component was to seamlessly retrieve data from model components during a simulation run and write them to the underlying PostSQL. Doing so enables modellers to view, edit, and manage simulation results using GUI of 3O-HIS. We need to notice that the location of shapefile in GUI should be same as the location in Geoserver which information stored in index.html. If it is consistent with location of map in web services, the Shapefile can be displayed in 3O-HIS. For example it is composed of geospatial vector data which represent triangular mesh in 3O-HIS in Blackpool (see Figure 5.6). The mesh representation in Network is simple that store the geometric data types of points, lines and polygons to represent geometric locations and usually linked to GIS. Points, lines and polygons have been used for creating Delaunay triangulation effectively. Each element of the triangular mesh may store many

physical parameters values.

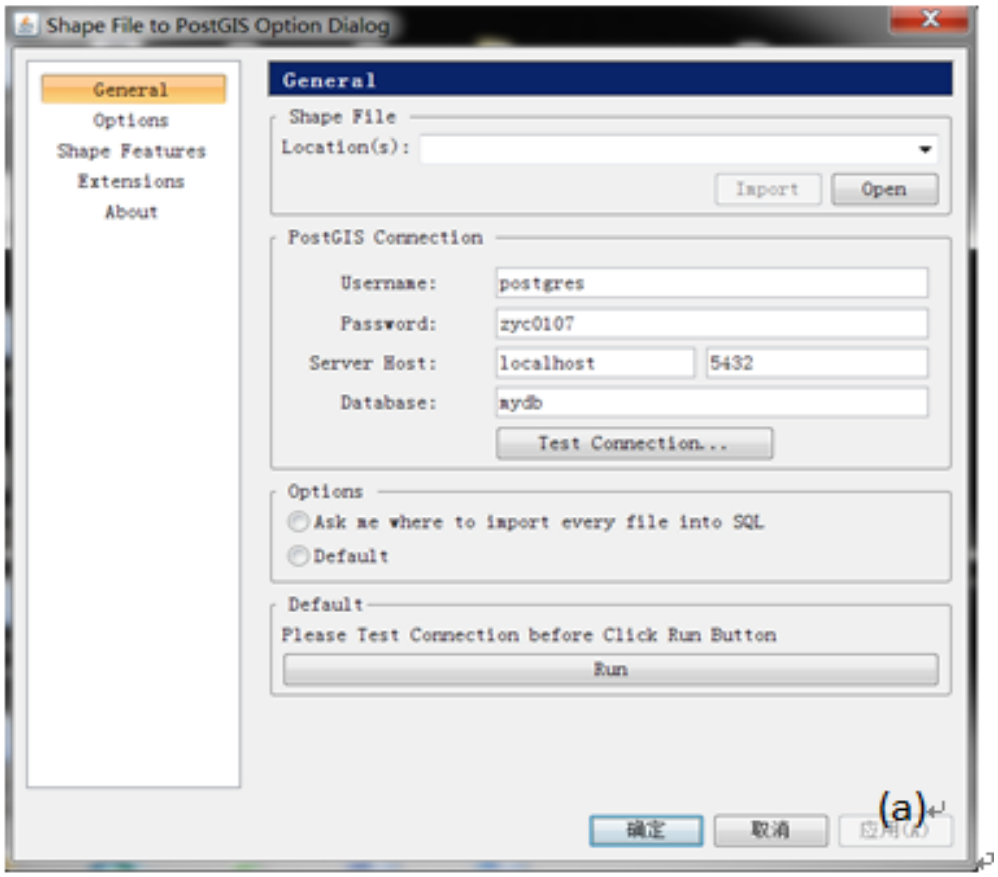


Figure 5.5 GUI of hydraulic model linking with 3O-HIS

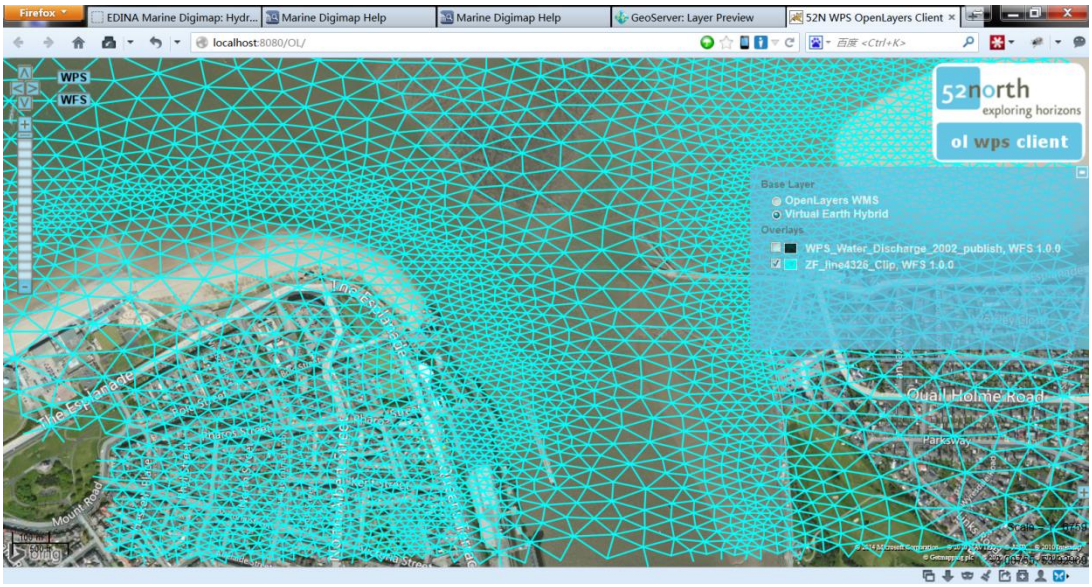


Figure 5.6 The retrieval of mesh of Wyre River Estuary Mesh

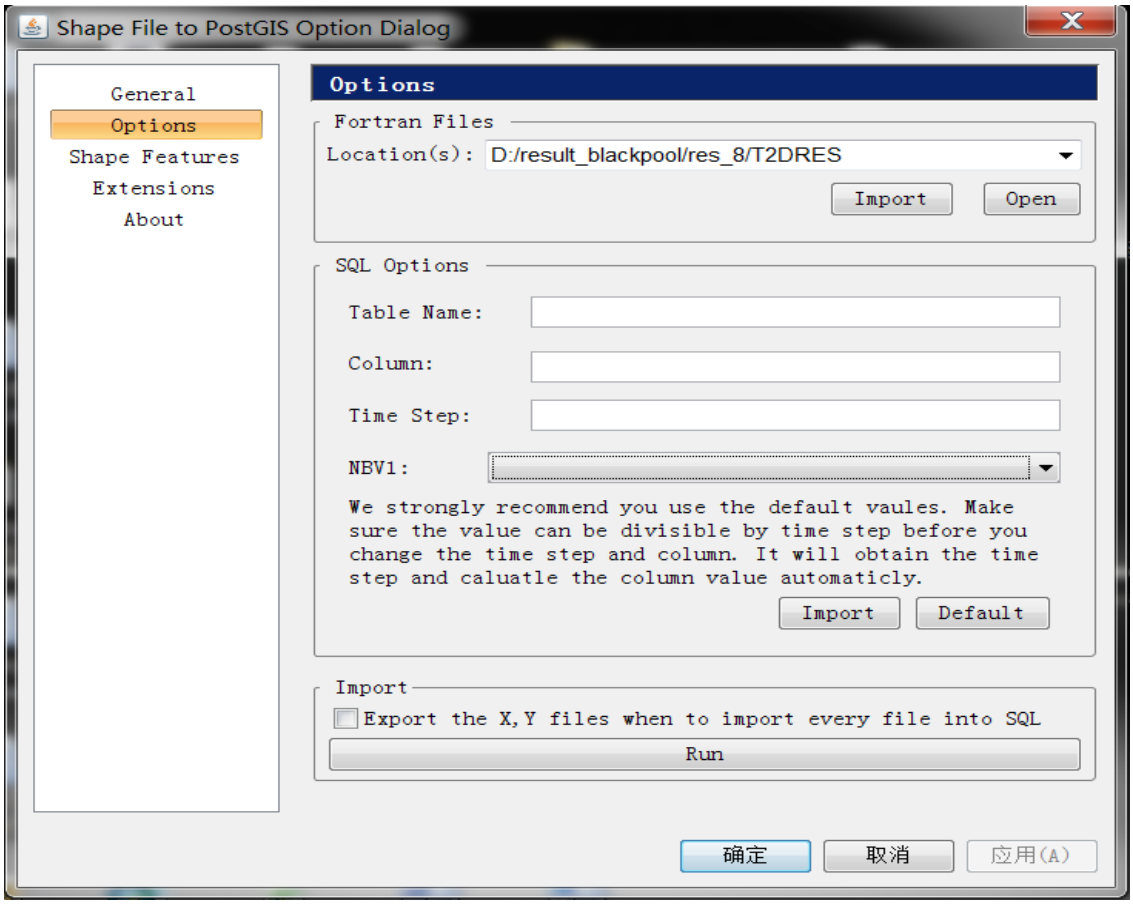
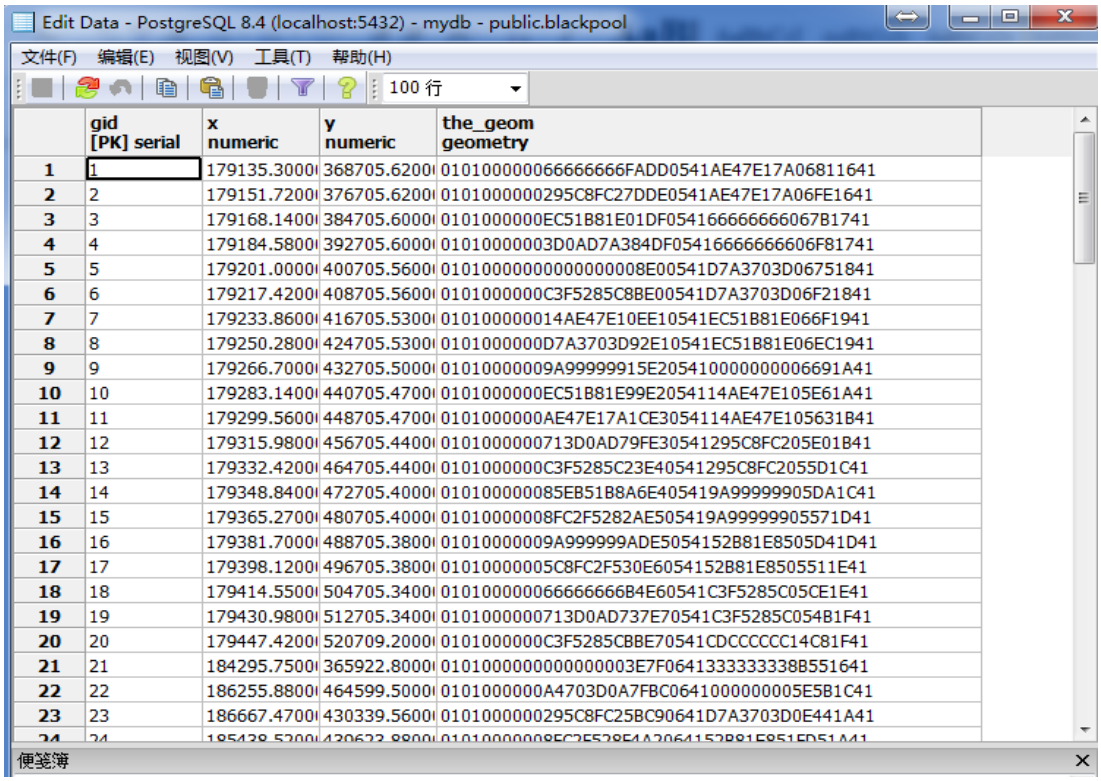


Figure 5.7 GUI for TELEMAC (Options tag)

In addition we created a default service which named ‘mydb’ with the username of ‘postgres’ and the password ‘zyc0107’. Figure 5.7 shows the setting of the parameters of transmission hydrodynamic data. The number of column of tables and the time step for hydraulic models can be settled in the GUI. Once click the ‘run’ button, the results storage in SQL (Figure 5.7). For linking PostSQL database, connection() function is used for

a PostSQL connect. SQL options function is used to set up the tables for constant or common standards. Connect() functions are developed in 3O-HIS, particularly standards/ rules established between uses. Once the spatial database has been created and value of x, y in each point on the boundary input into a spatial database is shown in Figure 5.8. The gid serial column represents BOUNDARY_COLOUR parameter and X, Y column represent X, Y coordinates in the Ordnance Survey National Grid reference system, and the_geom column introduces the spatial information. ArcGIS displays the boundary database and it is important to understand there is no need to add more information to identify the boundary type (open or close boundary). In TELEMAC model, boundary file gives the numbers to each boundary node to determine the boundary type. However, in 3O-HIS, because the spatial information has been added by PostGIS, which offers the application programming interfaces to perform spatial query and

analysis, boundary type has been identified for each boundary node. Therefore, ArcGIS can recognize the boundary type itself. For example, boundary node located at water levels defining open boundary node. In contrast, at the land boundary, the water level should remain zero all the time. In Figure 5.9 the open boundary nodes display as black colour and land boundary nodes show as cyan colour.



	gid [PK] serial	x numeric	y numeric	the_geom geometry
1	1	179135.3000	368705.6200	0101000000666666666FADD0541AE47E17A06811641
2	2	179151.7200	376705.6200	0101000000295C8FC27DDE0541AE47E17A06FE1641
3	3	179168.1400	384705.6000	0101000000EC51B81E01DF0541666666666067B1741
4	4	179184.5800	392705.6000	01010000003D0AD7A384DF05416666666606F81741
5	5	179201.0000	400705.5600	0101000000000000008E00541D7A3703D06751841
6	6	179217.4200	408705.5600	0101000000C3F5285C8BE00541D7A3703D06F21841
7	7	179233.8600	416705.5300	010100000014AE47E10EE10541EC51B81E066F1941
8	8	179250.2800	424705.5300	0101000000D7A3703D92E10541EC51B81E06EC1941
9	9	179266.7000	432705.5000	01010000009A99999915E20541000000006691A41
10	10	179283.1400	440705.4700	0101000000EC51B81E99E2054114AE47E105E61A41
11	11	179299.5600	448705.4700	0101000000AE47E17A1CE3054114AE47E105631B41
12	12	179315.9800	456705.4400	0101000000713D0AD79FE30541295C8FC205E01B41
13	13	179332.4200	464705.4400	0101000000C3F5285C23E40541295C8FC2055D1C41
14	14	179348.8400	472705.4000	010100000085EB51B8A6E405419A99999905DA1C41
15	15	179365.2700	480705.4000	0101000008FC2F5282AE505419A99999905571D41
16	16	179381.7000	488705.3800	01010000009A999999ADE5054152B81E8505D41D41
17	17	179398.1200	496705.3800	01010000005C8FC2F530E6054152B81E8505511E41
18	18	179414.5500	504705.3400	0101000000666666666B4E60541C3F5285C05CE1E41
19	19	179430.9800	512705.3400	0101000000713D0AD737E70541C3F5285C054B1F41
20	20	179447.4200	520709.2000	0101000000C3F5285CBBE70541CDCCCCCC14C81F41
21	21	184295.7500	365922.8000	0101000000000000003E7F06413333333338B551641
22	22	186255.8800	464599.5000	0101000000A4703D0A7FBC0641000000005E5B1C41
23	23	186667.4700	430339.5600	0101000000295C8FC25BC90641D7A3703D0E441A41
24	24	185428.5200	430623.8800	0101000000

Figure 5.8 SQL statement – shapefiles

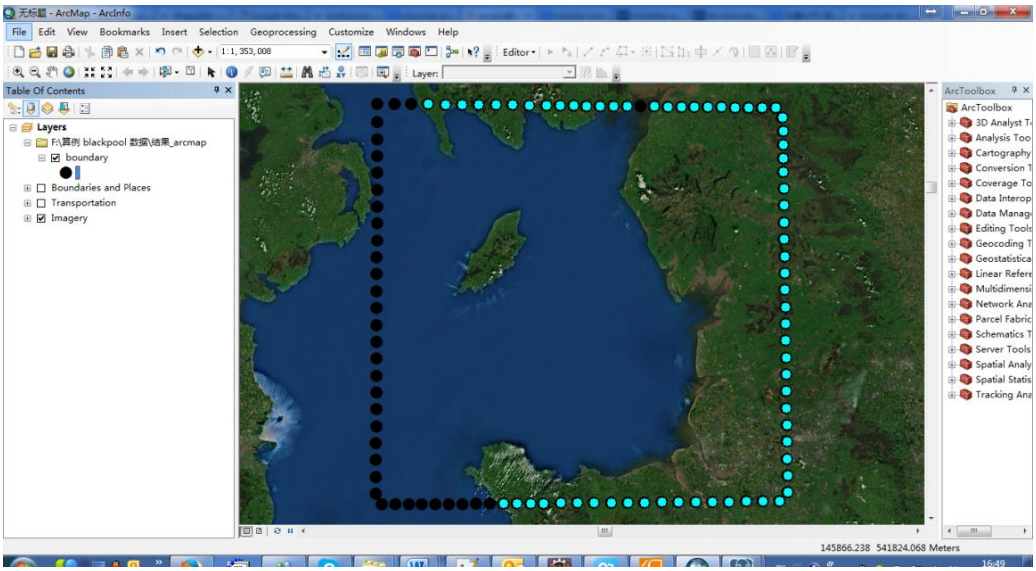


Figure 5.9 SQL statement – shapefiles

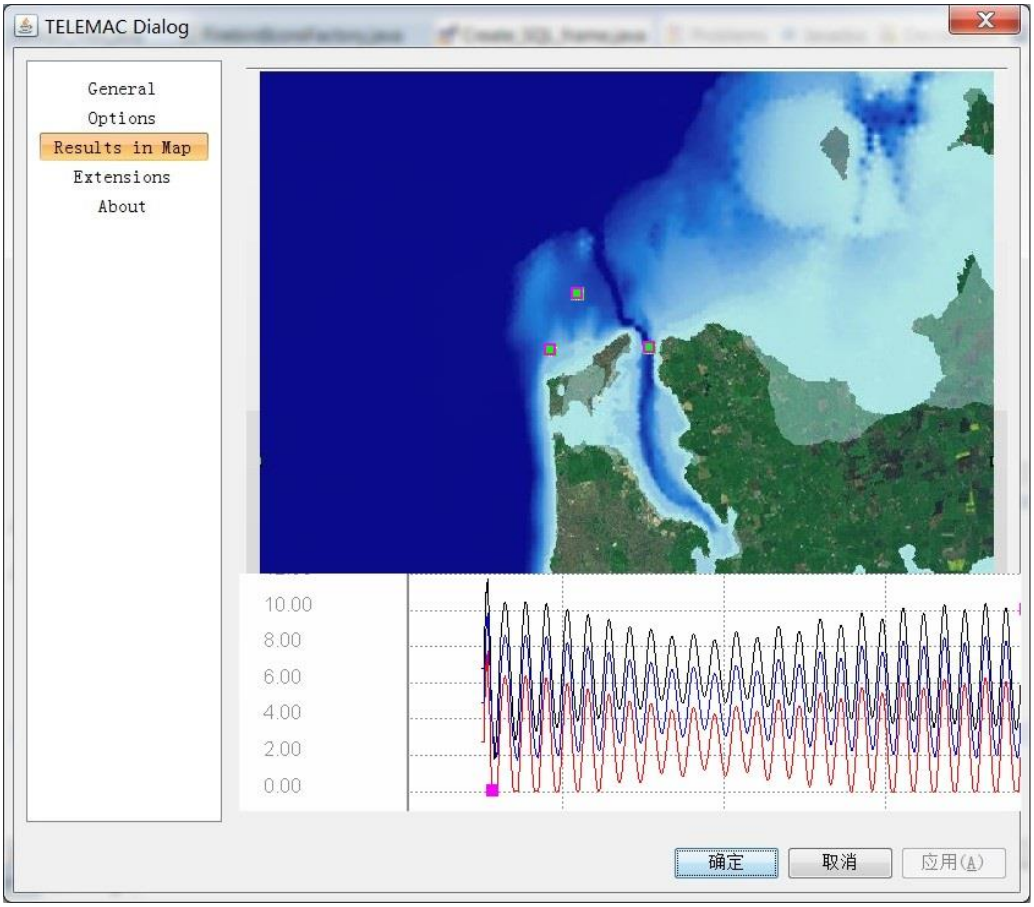


Figure 5.10 The inundation map and water depth with time series (three random points) in 3O-HIS

Raster file as background map is used to represent Geometry file. It is generated by a number of rectangular grids of pixels. Raster files in 3O-HIS, in essence, are digital image represented by grids. To enable an accurate flooding simulation, 10m x 10m raster graphics pixel has been used in 3O-HIS (Figure 5.10). Users can plot the time series at the interested location picked up in the digital map with the selected physical variable and specified time span through the GUI. The Figure 5.11 shows the sensitive degree of inundation that it extracts building layers related with water depth.

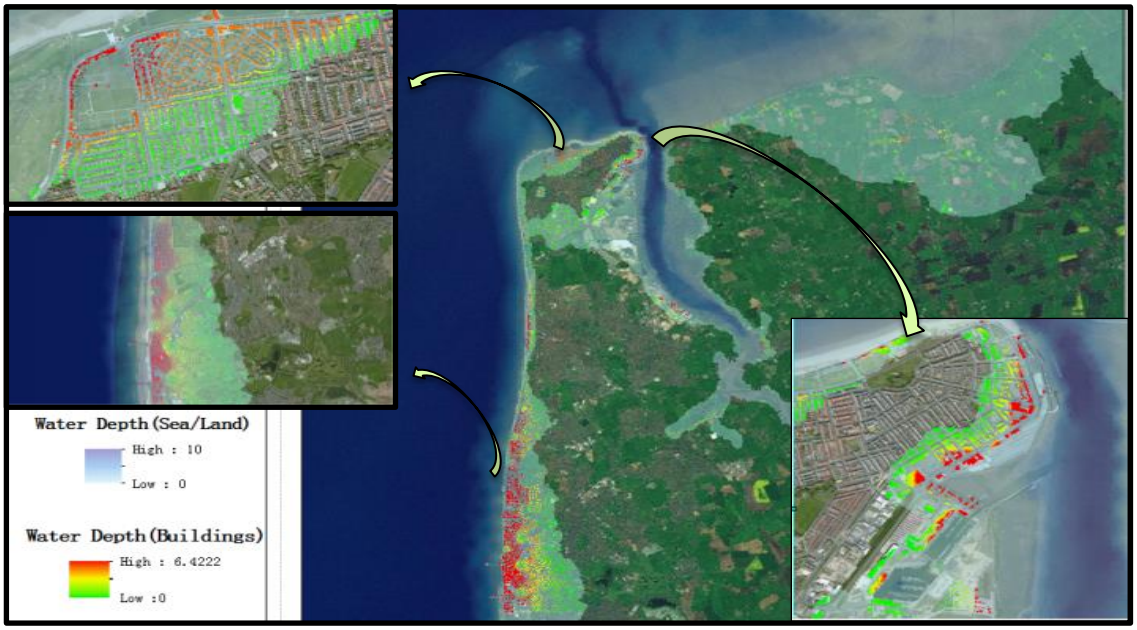


Figure 5.11 Water depths with time series

After the result map uploaded in service, GUI that it can retrieve the results, the Figure 5.11 shows the inundation map retrieved from web services. Comparing with single Telemac hydraulic modelling, the results include combing with GIS are a more representative way that provide an insight into the surrounding environment and associated physics in terms of prevented flood damage.

5.4 Service deploying for different hydrological models

3O-HIS needs link with other hydrological models to represent that 3O-HIS is an interoperable and open standard architecture. It is necessary to discuss the accessibility for linkage other hydrological models, and it can extend the range of hydrological models. To achieve the goal of expansion of applications, some configurations which need to be processed in hydrological models deployment are discussed in this section.

5.4.1 System configurations

From Figure of ‘the processing flow of 3O-HIS’ (see Figure 5.1, Page 102), different hydrological or hydrodynamic models are compiled and stored in ‘interoperability’ frame. It wills directly linage the ‘data storage’ component and ‘representation’ component. ‘Data storage’ component is the data service component to support the data communication and information sharing. ‘Representation’ component more related with data visualization represents the computational results displaying.

5.4.2 Service component configuration

Configuration of data service component is important because it is crucial step if the hydrological models changed and it is required re-definition the services components for input. Different hydrological models have varied type of date source as input. And different data sources storage properly in the service component is an important question if the hydrological models

changed. Vary type of dataset required carefully consideration for storage in services and interpolation between services under same standards. The comments have been represented for generally integrating other hydrological models.

- OpenGIS standards do not need to change. In 3O-HIS processing flow, the open standards are consistent from data services to representation component.

- It requires re-definition of Geometry file when hydrological models are changed. Checking the number of pixels in metadata files and the accessibility that new hydrological models load geometry file.

- Geometry file is needed as an input file to represent the topographical or bathymetric value and the associated mesh node locations where the calculation is carried out. The grids

can be reduced and enlarged and raster graphics pixel as the smallest individual grid unit building block of an image.

- Type of Boundary file usually is changed when new hydrological models applied. The information that the depth and velocity changed with time series at each point in boundary is used for driven the processing of simulation. Currently the file format supports TXT and Res.

- The design architecture of new hydrodynamic models requires consideration the input files before we deploy hydrodynamic models in 3O-HIS such as retrieval of hydraulic data sources from disparate repositories. The 3O-HIS's of framework flow should be consistent with the data processing, encapsulation web service and platform data viewing.

The next section will explain the Blackpool case study that simulates Telemac models to understand the Blackpool costal processing under various climate conditions are used to understand hydrodynamic models running in web service.

5.5 Case Study: Flooding forecast around Blackpool

The Blackpool to Fleetwood area in the North West England is a well-known low lying area with the mean surface elevation below the Highest Astronomical Tidal level and has always been susceptible to the coastal flooding (Department for Environment, Food and Rural Affairs 2010). In these areas, the total length of frontage is 24.4km and the existing flooding structures designed for a 1 in 200 year event calculated to protect a benefit area as 2,750 hectares (Wyre Borough Council, 2004).

Figure 5.12 Map of the Blackpool-Fleetwood area (After Wyre Borough Council, 2004)

This area includes 28,000 properties, 1,500 industrial units, 22.8 km of public highway, and 825 hectares of agricultural land. It has a high environmental value with the northern facing coastal

frontage from Rossall Point and the full length of the estuary being designated as a Special Site of Scientific Interest (SSSI) and candidate Special Area of Conservation (cSAC). The tangible discounted benefit derived from these assets is £165 million. Intangible benefits have not been considered numerically due to the potential to relocate tourist activities to neighbouring frontages. However, tourism plays a major role within the Borough employing over 3,200 people with estimated revenues of £60 million per year. The high value of the environmental assets and the tangible assets within the area has been determined (Figure 5.12), which are consequently related to the economic damage involved.

Since the recorded flooding event in 1891, there has been a number of severe floods occurred in the past, including the event in Year 1980 in which total 2000 hectares of land and over four hundreds houses were lost, and in Year 2000, extensive

flooding to the frontages of Ainspool area and Farms in the White Brook area. To be able to forecast the potential flood area due to extreme tide and river flows are very important to the management and sustainability of this area. The newly developed 3O-HIS system has been coupled with the coastal-ocean model of TELEMAC 6.2 (Hervouet and Bates, 2000) to formulate a new flood simulation tool and provide forecasts of the flood process under extreme tidal condition and the potential impacts and risk associated with various land uses in this area. The reason to choose TELEMAC system is two folds: at first, to enable accurate representation of flood in the low lying area due to tidal flows, the model should be able to simulate the water run up to the land the flood the originally dry area and then when the tide retreat to the sea (ebb), the model should be able to allow the flood area to become dry again, i.e. the capability to simulate the wet-and-dry process which is critical for the flooding simulation. TELEMAC has been

approved to be able to produce reliable predictions under the oscillatory flows over complex bathymetry with wet-and-dry process (Hervouet and Bates, 2000). Secondly, TELEMAC is an open-source code system which all source code can be accessed and hence it is much easier to modify the simulation or inputs-outputs of the system to combine with the proposed 3O-HIS system in comparison with many other commercial codes with similar capability. The following sections discuss details of the system configuration and coupling between 3O-HIS and TELEMAC model.

5.5.1 TELEMAC in 3O-HIS

Observation means the environmental data made by environmental observatories. For example, the data of water temperature are recorded by using in-situ sensor and the measurements of flow made using Price current meter in United States (Linsley et al., 1992). Recently the robust communication

infrastructure such as Internet connections, radio, satellite provide the availability of investigation of data at remote locations and more powerful for data digging. These technologies meet observatory data collection needs.

In TELEMAC model, a Steering file, a Boundary file, a Geometry file is required for the calculation. The Steering file and Boundary file are ASCII text files that can be readily created through any text editor, the Geometry files usually are binary and saved as GEO or SLF format. A Steering file has directives which are sending the main body of the TELEMAC modelling (Figure 5.13). Each line within the Steering file represents a command for TELEMAC to follow in the calculation. Usually many constant parameters such as friction coefficient, velocity diffusivity, initial elevation and time step are defined in the Steering file for TELEMAC simulation. In Boundary file, each line of the file is dedicated to one point on

the mesh boundary. Each point on the boundary can be specified a number that is very important for prescribing values. The bottom geometry file can be readily processed while the mesh is being built using MATISSE or BLUEKENUE (Telemac user manual, 2010).

```
/COUPLING WITH              ='inter-sisyphe'
/BINARY DATA FILE 1         ='merseydeewsteady.res'
RESULTS FILE               ='LbayTel.res'

/-----------------------------------------------------
/ INPUT-OUTPUT, GRAPHICS AND LISTING
/-----------------------------------------------------
LISTING PRINTOUT PERIOD          =300
INFORMATION ABOUT SOLVER         =YES
GRAPHIC PRINTOUT PERIOD          =300
VARIABLES FOR GRAPHIC PRINTOUTS ='U,V,H,K,E,W,D,S,B'
MASS-BALANCE                     =YES

/-----------------------------------------------------
/ INPUT-OUTPUT, INFORMATION
/-----------------------------------------------------
```

Figure 5.13 ASCII text file: Steering file

Depending on the simulation conditions, a FORTRAN file sometime is also necessary for the TELEMAC simulation. In the

present flooding case, a FORTRAN file is used to create the necessary water level information along all open boundary nodes at each time step so that the tide can be represented properly in this area. In particular, the instantaneous water level along the seaward boundary is prescribed as a sum of series sinusoidal function that is based on the tidal constitutions with particular range and frequency. Once the water level is given at each node, the computation within the domain can then be carried out based on mass and momentum conservation laws written in TELEMAC.

5.5.2 Telemac service bus in 3O-HIS

For clearly understand the mechanism of encapsulating TELEMAC model in 3O-HIS, the 3O-HIS service bus has been shown in Figure 5.14. It includes obligatory files as input files to drive the 3O-HIS, as well as Telemac model as a hydrodynamic model to calculate hydrodynamic characteristics, as exemplified

in Figure 5.14.

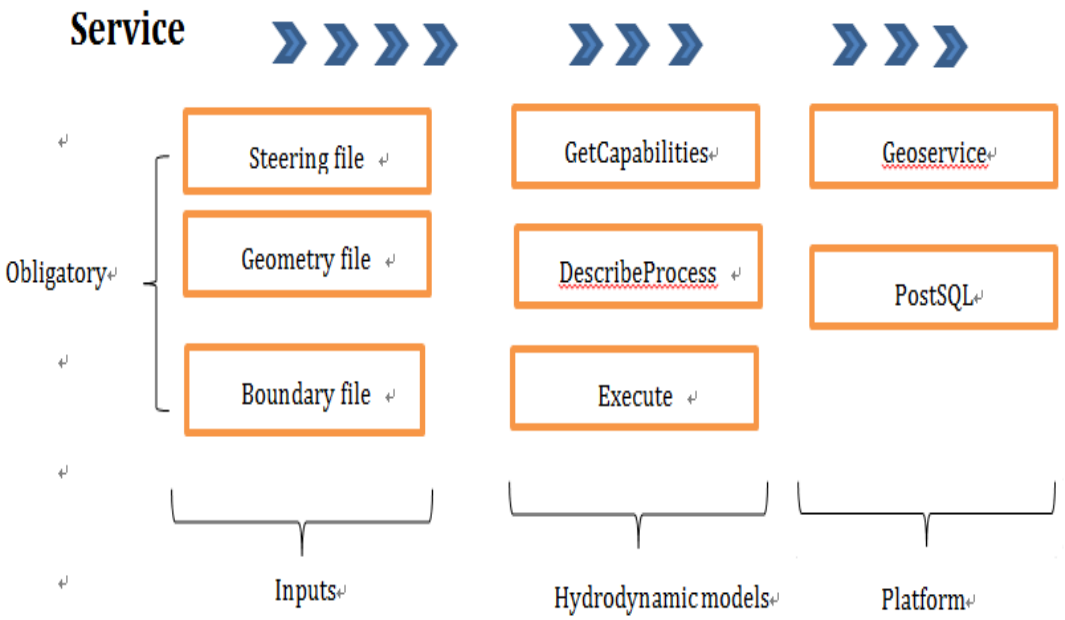


Figure 5.14 3O-HIS service bus(linkage TELEMAC model)

The mechanism of data transmission between services in 3O-HIS is similar to the client/server typical three-tier architecture. Different types of input files in Telemac require converting into an appropriate format which can be stored in services in a distributed manner. A lot of input data files are required and stored in different services such as Geometry service and Initial service call this process as ‘Inputs’. The files can send to ‘Hydrodynamic models’ in order to able to evoke

Telemac model. 3O-HIS architecture explained in chapter 4 had a facility for transporting input files such as three obligatory files-Steering file, Geometry file and Boundary file. Except three obligatory files, many other files can also interact in service which contains specific number of physical parameters during a simulation for specified conditions.

The advantages of GIS with hydrodynamic model in service orientation architectural style and its capabilities have been demonstrated by two case studies: Blackpool in England and Demra in Bangladesh. Two case studies are represented in order to: 1) demonstrate the advantage of GIS based 3O-HIS, comparing with single Telemac hydraulic modelling, combing GIS has the potential benefits of distributed collaboration; 2) illustrate some results during system configurations which are necessary when designing a distributed application in order to satisfy requirements such as identification of steering file,

geometry file and boundary files on both local service platforms in service bus.

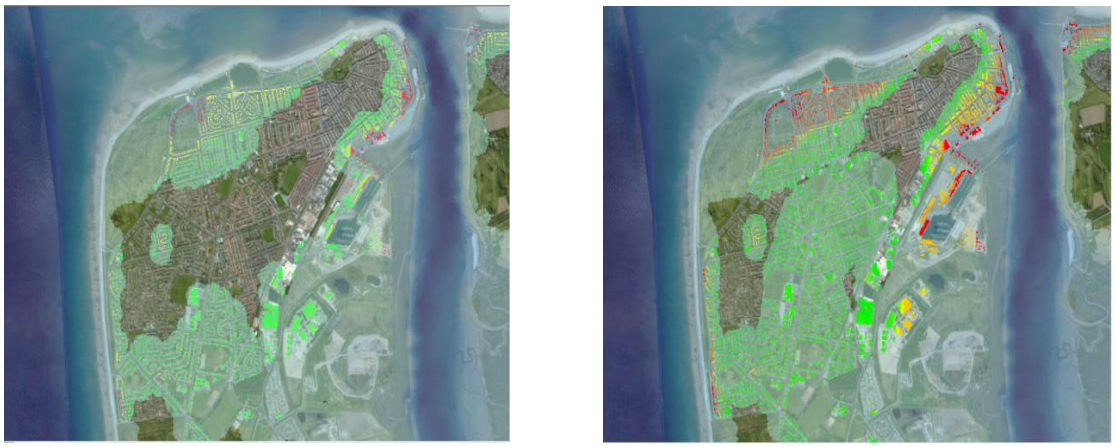
5.5.3 Flooding forecast data analysis

The case study of Blackpool shows the multi-object analysis in 3O-HIS, and it can extend the Telemac simulation capability: land use, sea level rise and socio-economic factors.

The main problems in this area with the estuary are their ad hoc nature, lack of maintenance and in some instance the presence of weakness and low defences. Future sea level rise will increase coastal squeeze within the estuary leading to loss of protective salt marsh frontages and direct erosion of the flood embankments.

The Intergovernmental Panel on Climate Change (IPCC) estimated a global sea level rise with a ‘best-guess’ value of 49

cm by the year 2100 (Gornitz, 1994). The potential subsidence issues reported by Blackpool Borough Council could substantially increase the apparent SLR. Two sea-level rise scenarios for the year 2100 have been adopted for the present analysis: 'best-guess' (SLR 49 cm) and High (SLR 88 cm) which are derived from IPCC and Gornitz (1994). A failure in the upper estuary defences could lead to flooding to the majority of the urban area, and particularly in the low-lying areas which is shown in Figure 5.15.



(a) (b)

Figure 5.15 a) inundation map SLR 0cm in Fleetwood, b) inundation map SLR 49cm in Fleetwood

This is the most vulnerable area under the SLR 49cm scenario;

almost 2/3 of the area has been inundated. Two costal lanes which we need to carefully consider are Fleetwood Pier to Rossall Point and St Mary's R C Church to Fleetwood Museum. The first one is left corner of northern facing frontage, most of the houses affected immediately if the upper portion of sea wall fails. In the second vulnerable area, St Mary's R C Church and Fleetwood Museum buildings may be seriously affected due to 49cm sea level rise.

Table 5.1 Inundation areas based on the different sea level rise

	Flood depth(m)	Area m^2(SLR 0cm)	Area m^2 (SLR 49cm)
Urban area	0-0.5	28452000	31341600
	0.5-1	4095100	7912100
	1-2	3894700	5035000
	2-3	340100	698100
	3-4	24700	52600
	4-6.42	21000	33300

Direct physical damage related to flood depth to derive based on empirical flood damage data from two catastrophic floods- in 1953 in the Netherlands and in 1993 river Meuse. Later flooding events the Elbe flood in 2002 and the New Orleans in 2005

provides important updates and specific damage factor is estimated and relates to flood depth.

The total inundation area is 36827600 m^2 (SLR 0cm). Major flood depth is below 0.5 m and 77.25% of the total inundation area. Under the SLR 49cm scenario, the total inundation area is 45072700 m^2; it increased by 22.38% compared to the SLR 0cm as shown in Table 5.1.

From Table 5.2, it is nearly 0.7 billion increased in sea-level rise scenarios for the year 2100. Currently, the urban area and households are distinguished in the direct physical damage assessment. Urban area has a high density of population and more vulnerable in terms of human and economic losses. Site-specific conditions or regional differentiation are not accounted for in the assessment procedure such as agriculture.

Table 5.2 Total Direct damage amount based on the different sea level rise

Damage category	Flood depth(m)	Measurement Value	Damage factor	Per damage amount	damage amount	Total damage amount
Urban area	0-0.5	28452000	0.1	4.9	139414800	2.73billion
	0.5-1	4095100	0.15	7.35	30098985	
	1-2	3894700	0.2	9.8	381680.6	
	2-3	340100	0.35	17.15	5832715	
	3-4	24700	0.6	29.4	726180	
	4-6.42	21000	1	49	1029000	
Households	0-0.5	65851	0.1	24100	1587009100	
	0.5-1	10577	0.15	36150	382358550	
	1-2	10113	0.2	48200	487446600	
	2-3	877	0.35	84350	73974950	
	3-4	103	0.6	144600	14893800	
	4-6.42	48	1	241000	11568000	
Urban area	0-0.5	31341600	0.1	4.9	153573840	3.42billion
	0.5-1	7912100	0.15	7.35	58153935	
	1-2	5035000	0.2	9.8	77538580	
	2-3	698100	0.35	17.15	11972415	
	3-4	52600	0.6	29.4	1546440	
	4-6.42	33300	1	49	16317	
Households	0-0.5	70861	0.1	24100	1707750100	
	0.5-1	17371	0.15	36150	627961650	
	1-2	11278	0.2	48200	543599600	
	2-3	2403	0.35	84350	202693050	
	3-4	73	0.6	144600	10555800	
	4-6.42	106	1	241000	25546000	

Consideration of the regional differentiation undoubtedly helps to understand flood characteristics and transfer flood risk and flood damage assessments across different scenarios. Further, careful consideration and examination of land use can make damage estimations more reasonable. For the results, we can summarize that beach volumes are consistently improving; consideration on maintaining and repairing existing structures and examination of the need to provide improved crest levels where defences are weak. It needs a comprehensive estuary shoreline management policies and boundary definitions.

5.6 Case Study Area: Demra of Bangladesh

Dhaka, the capital of Bangladesh, is situated in the central part of the country and has an area of 298 Sq. Km. It is bounded by the Buriganga River in the south, the Balu River in the east, the Tongi Khal in the north and the Turag River in the west (see Figure 5.16). These rivers are connected to the

Ganges-Brahmaputra River system (locally known as the Padma-Meghna-Jamuna River system, which also includes the Old Brahmaputra River) that flows southeast from all sides of the larger neighbouring study areas. The larger area is closely dissected by a number of rivers and canals that are connected to these major rivers. Dhaka is the fastest growing mega-city in the world (Alam and Rabbani, 2007), with an estimated 300,000 to 400,000 new migrants, who are mostly poor, arriving to the city annually. Its population is currently near 14.2 million and is projected to grow to 20 million in 2020, making it the world's third largest city (UN 2008; UN Population Division, 2010).

The landform of the city is characterised by the Madhupur Tract, an elevated Pleistocene terrace (Morgan and McIntire, 1959) that stands higher than the neighbouring floodplain and low-lying marshlands.

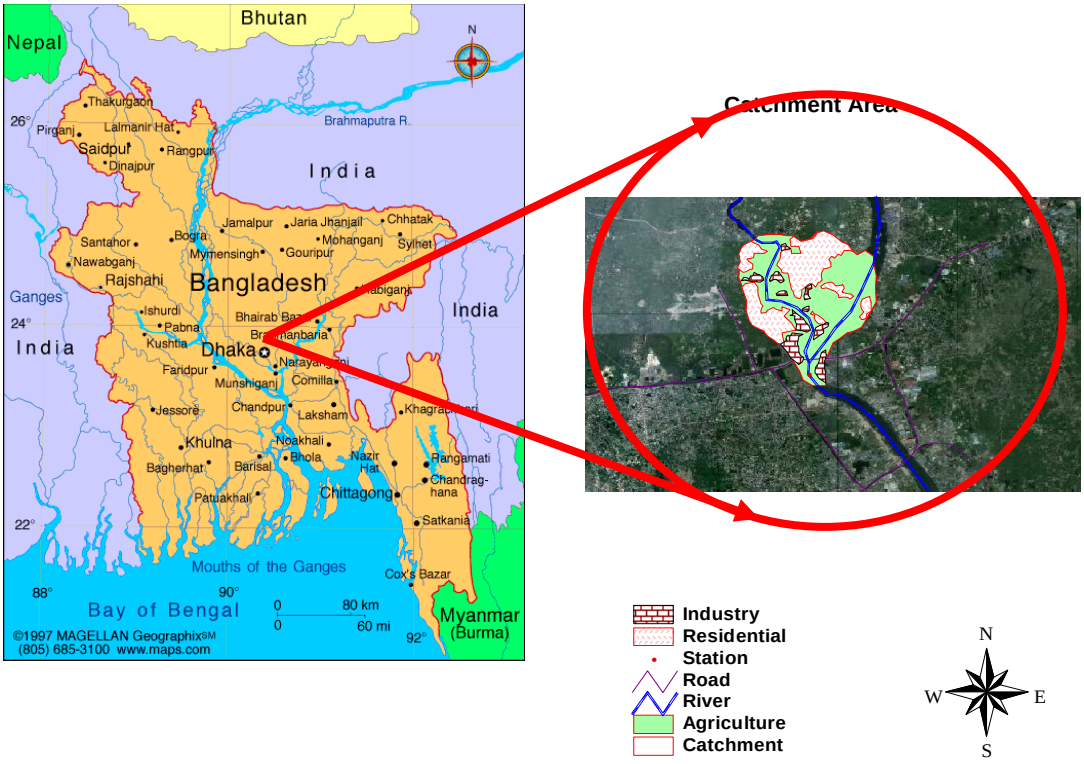


Figure 5.16 Location Map of the Study area (Source:Magellan Geographix)

The area represents a significant variation in elevation ranges from 1.5 to 15 m, with an average of 6 m above PWD (Public Works Datum) (+/- 0.45 meter with respect to mean sea level). The area slopes towards the southeast, east and west, but the general slope extends from the north to the southeast, where the ground surface merges gently with the floodplains of the Buriganga River. The eastern edge is mainly covered by the

floodplains of the Balu River. These floodplains are characterised by low-lying depressions and marshy areas that remain inundated for a significant period of the year. The storm runoff accumulates in the low-lying areas, flows through canals and local rivers and slowly discharges to the major rivers. These lowlands and wetlands are performing important drainage functions by storing storm water and keeping the relatively higher lands free from rainfall flooding. The area has a tropical monsoon (May to October) climate like other parts of the country, with an average precipitation of 2000 mm/year.

5.6.1 Data definition/ Collection

The GIS data were used in the form of vectors or shape files to delineate the catchment or river basin areas. Collection of the vectors is similar to the collections represented in section 5.4. Different shapefiles represent the different land use in the catchment Area.

The variable parameters are different with the variable parameters in Telemac, in the water balance model, the various climatic data are the precipitation in mm, mean maximum and minimum temperature (℃), air humidity (%), wind speed (km/day) and daily sunshine (hours). All these data can be obtained from the Bangladesh Meteorological Department (BMD). The precipitation in mm from the year 2002 to 2011 has been stored in a web service in this case study in Figure 5.17. We can compare the precipitation data from BMD with the data calculated in 3O-HIS.

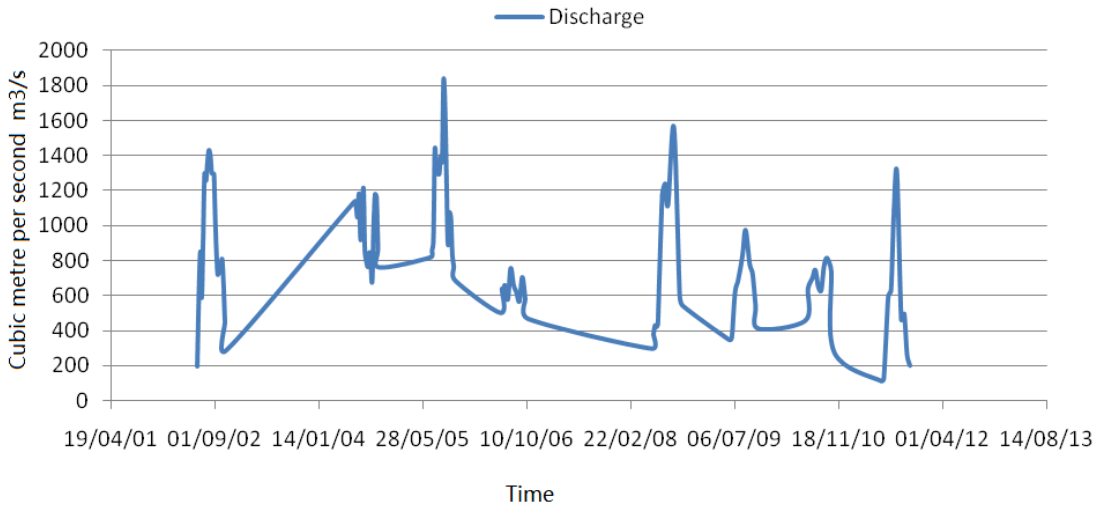


Figure 5.17 Discharge of Lakhya river from 2002-2011

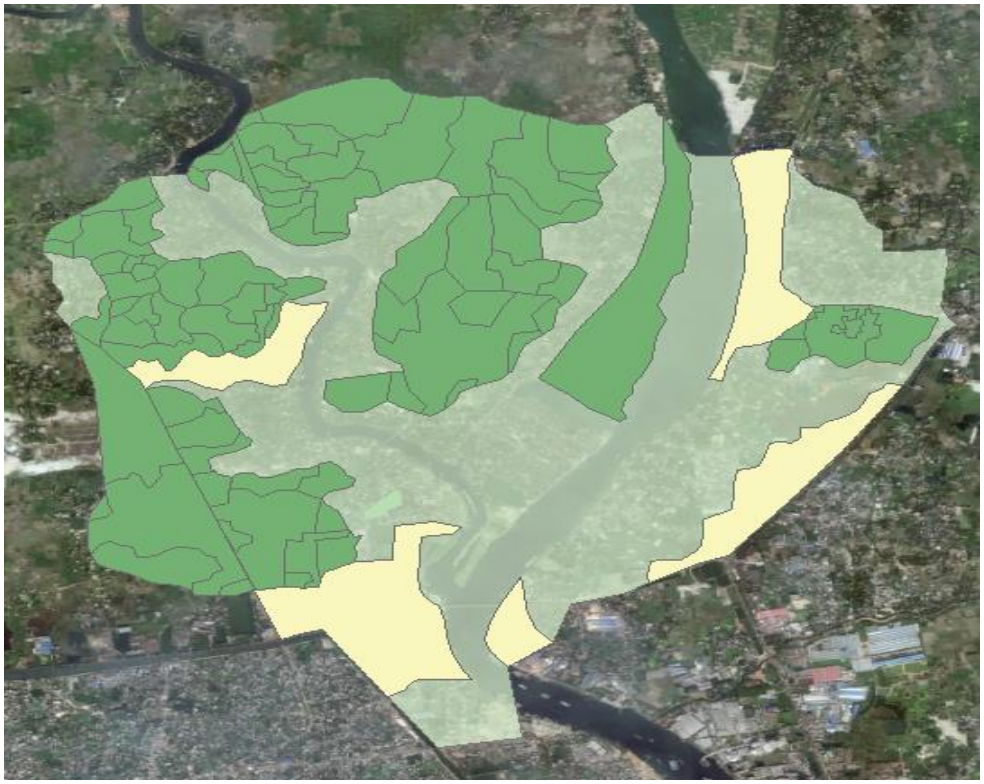


Figure 5.18 Land use in GIS ArcView

The agricultural areas are shown in Figure 5.18. Each agricultural area was calculated using a spatial function of GIS ArcView, The catchment area of 37.69 sq. km. was calculated using the spatial function of GIS ArcView. Of this area, 12.62 sq. km is a residential area and 3.37 sq. km. is an industrial area. The rest of the area (21.7 sq. km) is agricultural and it has been divided into 74 agricultural areas which have a minimum value of 90.488 m^2 and a maximum value of 489392.728 m^2. Based on

the topological information, discharge of each area is calculated via the Eq. (1):

$$\text{Discharge}_i = \text{Total Discharge} \times (\text{Area}_i / \text{Total area}) \quad \text{Eq. (1)}$$

Table 5.3 Values of K for selected crops (Source: Subramanya, 2008)

Crops	Average value of K	Range of monthly values
Rice	1.10	0.85-1.30
Wheat	0.65	0.50-0.75
Maize	0.65	0.50-0.80
Sugarcane	0.90	0.75-1.00
Cotton	0.65	0.50-0.90
Potatoes	0.70	0.65-0.75
Natural Vegetation		
Very Dense	1.30	
Dense	1.20	
Medium	1.00	
Light	0.80	

The developed water balance model, two values: coefficient K and N are important in predicting the discharge. Values of K is variable based on the crops, and based on the topological information, the crop coefficient K of 74 agricultural areas are valued in the catchment in Bangladesh as shown in Table 5.3.

Coefficient N used to represent the difference between the predicted and measured discharges. The Relativity parameter N used to represent the difference between calculated/predicted and measured discharge is defined as follows:

$$N_i = 5\sin(\frac{\pi}{2} \times |Prec - MPrec| \times K) \ldots\ldots\ldots\ldots\ldots\ldots \text{Eq. (5)}$$

Ni represents the relativity between the predicted and measured discharge. The Ni value usually does not exceed 5. A low Ni value suggests that the predicted discharge closely approximates the measured discharge. K is a random value between 10% and 15% based on the type of crop.

5.6.2 Results and Discussions

In Demra case study, the water balance model is developed and the hydraulic modelling is entirely written by Java code. The data transmission is directly through services under OpenGIS protocols.

The response time of water balance model in 3O-HIS has been compared with the running time of the water balance model on local service. The response time of the web service was roughly identical to that of the local PC in all 141 tests. Once the map data were downloaded from a web service, they were accessed as local service. Therefore, the response time of the web service performed better than the local calling process, as shown in Figure 5.19. Moreover, both services could retrieve data fewer than 100 ms in most cases.

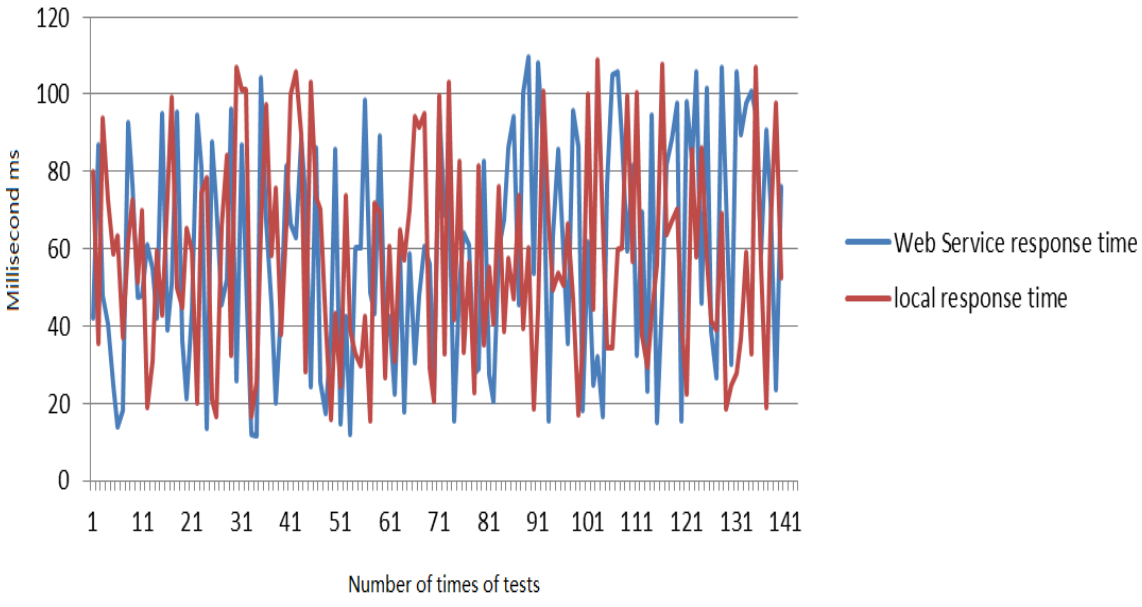


Figure 5.19 Web service and local response Time with 141 times test

The results suggest that the 3O-HIS architecture and distributed infrastructures can offer solutions to the challenge of interoperability for data collected at a low level (72 agriculture areas as a data input). Due to the lack of data, the web service response time was not compared with the local response time in the present study. Goodall (2008) proposed that the response time significantly increased since the date when the number of records exceeded 10,000. Because the same XML technology was used for service communication in the present study, the response time may be affected when large amounts of data are collected.

The accumulation of large amounts of data invariably increases the response memory size due to encoding objects in GML. However, 3O-HIS provides an optimised architecture, i.e., service chaining and reuse, which reduces objects in GML to minimise the problems of data accessibility and interoperability

in distributed services.

5.7 Comparing uDig-Telemac with ArcGIS-Telemac

The discussion concentrated on the service orientation and system configurations and Open source GIS has been discussed the processing flow between services and many advantages of the deployment of hydrodynamic models using Open source GIS such as interoperability and extendibility.

Flooding simulation around Blackpool has been carry out using ArcGIS-Telemac, To represent that 3O-HIS is an interoperable and open standard architecture to support large amounts of commercial and free GIS software, it is necessary that carrying out one simulation for the flooding forecast using freely open standard GIS software.

Unlike commercial GIS, Open source GIS is more widely used

by “government, businesses, and non-profits alike because of the financial benefit” (Confino and Laplante, 2010). UDig is one of the most popular open source GIS software, built with Eclipse Rich Client technology, an example of lining uDig with Telemac has been shown in Figure 5.20. In the rest part, a compression is discussed between ArcGIS-Telemac and uDig-Telemac to demonstrated that 3O-HIS can support Open source GIS seamlessly.

Background of Blackpool has been described in section 5.4. Therefore, it will not be repeated here, and the system configurations are listed below (Table 5.1):

Table 5.4 System configurations between Tele-Arc and Tele-uDig

	Telemac-ArcGIS	Telemac-uDig
System environment	Visual Studio 2010	Eclipse
Geometry file	PostGIS	WMS
Boundary file	Notepad++	Notepad++
Parameter file	Geoserver	PostSQL

Telemac-ArcGIS and Telemac-uDig are under the same 3O-HIS

architecture, however comparing the system configurations (Table 5.1), most of the input files are processing under different software. Visual Studio 2010 is running for ArcGIS and uDig is processing under Eclipse, Geometry file requires transmission and storage at PostGIS, and in uDig WMS services can be directly used for receiving geographical maps. In ArcGIS the Parameter file storage in service-side and the Parameter file stores in client-side in PostSQL service in uDig.

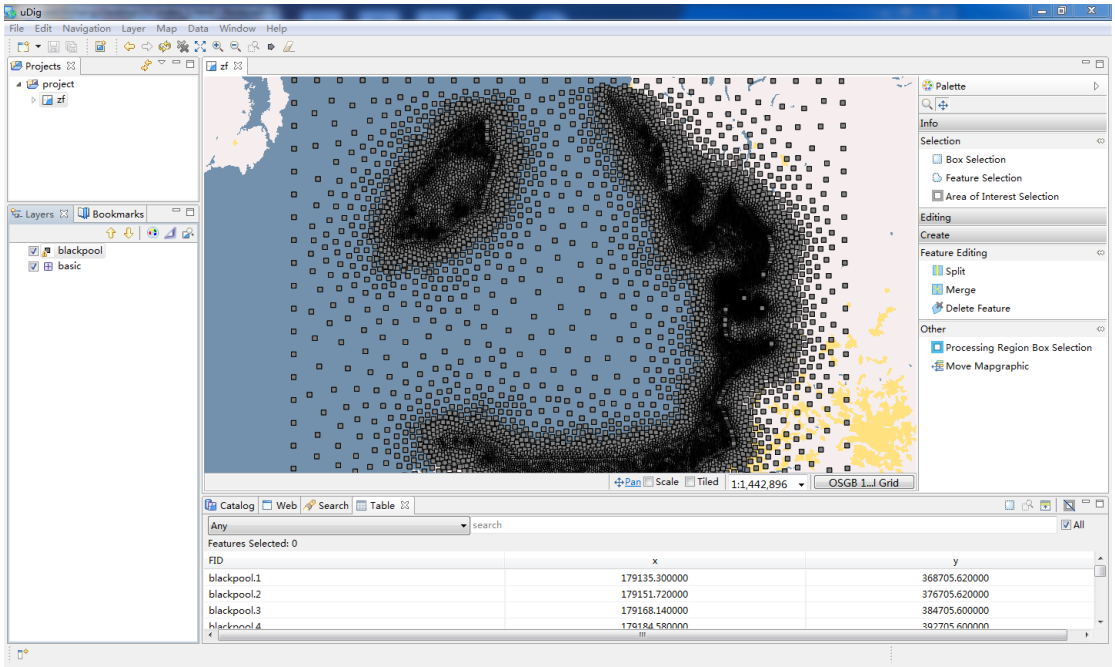


Figure 5.20 Blackpool features in uDig

Comparing with the Blackpool features in ArcGIS; there are

several differences with the features loading in uDig.

- The background map is the VirtualEarth WMS service in uDig with the coordinate reference system ‘4326’ code and the map in ArcGIS are downloading from VirtualEarth Hybrid with the coordinate reference system ‘27700’ code.

- There is no raster interpolation function in uDig which can generate water depth map with time series from points using an inverse distance weighted technique.

The results of response time from testing the basic functions shown as Table 5.5.

Table 5.5 Comparing the response time between Tele-Arc and Tele-uDig

	Telemac-ArcGIS	Telemac-uDig
Buffer	3.4s	0.2s
Clip	3s	0.12s
Select	2.2s	Less 0.01s
Split	3.6	0.15s

The results is an average value from 10 groups, each group randomly select 300 points in Blackpool mesh for buffer, clip,

select and split analysis. The results show that Telemac-uDig has a really fine response time comparing with Telemac-ArcGIS. It extremely likely that the system optimization. It means that, Eclipse, as a platform supports uDig, same as the platform for developing the code of linking hydraulic models and GIS. The consistency which exists at the bottom of infrastructure improves the performance of data analysis.

Combining with open source GIS tools, ArcGIS-Telemac still is a reasonable approach. It can be more powerful for spatial data analysis. The Telemac results can be projected to an appropriate coordinate system by using ‘Projections and Transformations’ functionality. Moreover, raster interpolation function can interpolate a raster surface from the points which are the results of Telemac model. For example the user can access and manipulate information associated with geographic features and look for spatial or temporal patterns and/or relationships in

different layers of the GIS map. In addition, a raster of accumulated flow into each cell is created in flow accumulation function in ArcGIS. The abundant function for data spatial analysis leads to achieve the gold of hydrology analysis using.

5.8 Summary

The association between SOAs under 3O-HIS brings newly ideas for developing modelling system and helps developers analysis hydrological data in a comprehensive way. By using these benefits, 3O-HIS is applied in two case studies to illustrate an newly and comprehensive way for hydrological data analysis over the conventional GIS system.

The different types of vector are used in two case studies. In the Blackpool case study, the mesh is created by a number of points and lines. The prediction of water level depth is stored in each point. The points represent the stations in National scale. In

Demra case study, to predict the discharge depend on the value of agricultural areas. The water balance model requests polygons uploaded in services.

The 3O-HIS was designed to explore methods to interoperate hydrologic data between servers' and users' computers. Two hydraulic models were deployed on servers; designed to be a service-oriented web application tool that makes water storage calculation on the Internet available to users by allowing them to access data and relevant web services in a distributed network.

Results obtained from 3O-HIS, helps policy makers to encompass the full management unit. It is the powerful ability of spatial data analysis to spatially distribute data and improve the resolutions of water resources problems. In addition, the network structure provides assistance to the policymakers on multiple sources of hydrologic data and hydrologic modelling as

services were encapsulated in the architecture of service layer of 3O-HIS. It supports Open source GIS and commercial GIS and increases the broad range of GIS and has the capacity of further expansion of applications and solving problems.

Chapter 6 Conclusions and Further works

6.1 Introduction

A new 3O-HIS is developed to encapsulate distributed hydrologic modelling for the purpose of interoperation in the integrated hydraulic applications. It provides a practical way to access hydrodynamic models in a web service environment. 3O-HIS is designed to explore methods to interoperate hydrologic data between servers' and users' computers. Hydrodynamic models, designed as a service-oriented web application tool was deployed on a server which makes hydrologic applications on the Internet available to non-expert users by allowing them to access data, process them and publish results in a distributed network.

Comparing with other traditional framework–GIS based approaches, 3O-HIS has a high level of performance which is designed under the concept of loosely coupled self-contained services and is developed for optimal, scenario exploration and decision support. It provides a flexible platform, loosely coupled self-contained services, data accessibility and service interoperability for environmental models.

6.2 Major findings

3O-HIS, a common framework has been developed to provide an innovative way to make full use of existing and emerging standards to aggregate heterogeneous geospatial data and observational data, and deliver rich user interface for users to perform domain analysis tasks at client-side. In particular, the new 3O-HIS system includes the following components: hydrological models, data processing based on OpenGIS protocols, GUI for data representation.

6.2.1 Components of 3O-HIS

Hydrological models are the fundamental component in 3O-HIS. The models can be integrated and comprise of an ‘engine’ for hydrodynamic data generator. The validity of the hydrological models mainly depends on the selected hydrodynamic data and it is out of scope. Our concern is how hydrologic models encapsulated smoothly in 3O-HIS representing by two case studies. In Blackpool case, the process of encapsulating TELEMAC model into 3O-HIS has been represented. In Demra case study, map information, physical parameters and water balance model are uploaded into 52n service. Shapefiles and PostSQL in both studies can be loaded in GIS directly and the data transmission depends on OpenGIS standard protocols such as SOAP, XML and GML.

Simplicity is the key word for describing the data processing flow of 3O-HIS, we try to make data conversion simply during

the data processing. Integration these specific standards in different hydrological models, users may have more concise way to select hydrological models, it is necessary to develop a more general way of encapsulating different models. Currently, many common file formats can be converted such as TXT, HTML, and dbf and most of raster files such as jpg and bmp. For example, In Demra case study, jpg files have been used for raster date analysis where each pixel represents 10m ×10m areas, the value of agriculture areas.

The concise solution requires a series of protocols which are implemented during data processing. XML and GML are two key protocols for data transmission such as physical parameters, information of messages in services (call/response). Both of them are tag pairing, and the information is stored in each pairing. The pairing method extends the types of data resource stored in 3O-HIS. To ensure the pair of tag are same, and there

is no constraint for the information between the tags.

A GUI has been developed for data representation and the data can be represented in common GIS applications which allows link to database. For example, a GUI has been created and links to PostSQL for directly checking the results in the Blackpool case and the GUI links to 52n services in Demra case to represent data interoperation.

6.2.2 Application of 3O-HIS

Two cases have been used in 3O-HIS. In the Blackpool case study, the bottom file, and boundary file convert into shapefiles and PostSQL database stores physical parameters. While in Demra case study, polygon maps and raster maps are loaded in 52n services. Two case studies represent the hydrological models distributed to local and services. Four major findings can be found:

- 3O-HIS runs TELEMAC model initially on local, huge of hydrological data are generated (about 6.5 GB), and we reorganized the structure of results in TELEMAC from representing all hydrological value in each time to represent each hydrological value in the time series. After the tests, when the time step=3 hours, it is most efficient for container carries hydrological value.

- GML2 is used for geometry information exchange. GML2 has the function of gml:posList which can present the geometry files such as mesh, XML has no specific instructions for geometry file. GML2 is recommended for geometry information of message exchange.

- Small pieces of hydrological models are more suitable than the single frame hydrological models. The water balance

model is small one, which is deployed on WPS services directly. The TELEMAC haven't been encapsulated in services. The major reason is speed, we test the data exchange, it shows that the common time cost is 0-1s for each send/receive processing, the iteration in TELEMAC arithmetic is huge and time consuming will be considerable.

- Comparing with the original CUAHSI HIS, the 'HIS centre' which is a vital part in CUAHSI HIS replaces several protocols. We used XML, GML and SOAP for data exchange. It is concise for data exchange. However, the time response is longer than CUAHSI HIS.

6.3 Further works

- Grid computing with service-orientation structure, to processing computational hydraulic applications in internet-scale. Currently TELEMAC integrate with 3O-HIS

at a low level, Grid computing with service-orientation provides opportunity for conglomeration other services, using idle processing power, have a high performance computing for computational hydraulic models.

- Balance the granularity of services, one solution from the major fining is that TELEMAC are divided, and refined services compose the hydraulic model. The granularity of each refined service need to careful design. Too coarse will reduce the flexibility, Too refined will reduce the computational performance.

- Balance the protocols and the services. From last finding shows some services are redundant for storing massive data to enhance the performance and protocols may control the behavioural services.

References

Alam, M., Rabbani, G., 2007. Vulnerabilities and responses to climate change for Dhaka, Environment and Urbanisation, IIED, 19 (1), 81–97.

Alonso, G., Casati, F., Kuno, H., Machiraju, V. 2004. “Web Services: Concepts, Architectures and Applications”, Springer-Verlag, Berlin Heidelberg New York.

Al-Sabhan, W., Mulligan, M., Blackburn, G.A., 2003. A real-time hydrological model for flood prediction using GIS and the WWW. Computers, Environment and Urban Systems, 27, 9-32.

Ames, D.P., Horsburgh, J.S., Cao, Y., Kadlec, J., Whiteaker, T., Valentine, D., 2012. HydroDesktop: Web services-based software for hydrologic data discovery, download, visualization, and analysis. Environmental Modelling & Software, 37, 146-156.

Argent, R., 2004. An overview of model integration for environmental applications: components, frameworks and semantics, Environmental Modelling & Software, 19 (3), 219-324

Argent, R., 2005. A case study of environmental modelling and simulation using transplantable components, Environmental Modelling & Software, 20 (12), 1514-1523.

Bain, D.J., Hale, R.L., and Wollheim, W.M., 2012. Hotbeds of biogeochemical diversity; insights from urban long-term ecological research sites, Mineralogical Society of America and

Mineralogical Society of Great Britain and Ireland and Mineralogical Association of Canada and Geochemical Society and Clay Minerals Society, International, 435-438.

Band, L., Patterson, P., Nemani, R., Running, S., 1993. Forest ecosystem processes at the watershed scale: incorporating hillslope hydrology, Agricultural and Forest Meteorology, 63 (1–2), 93–126.

Bates, P.D., Roo, A.D., 2000. A simple raster-based model for flood inundation simulation, Journal of hydrology, 236 (1), 54-77.

Battjes, J.A., Gerritsen, H., 2002. Coastal modelling for flood defence. Philos Trans A Math Phys Eng Sci. 360(1796),1461-75.

Beran, B., Piasecki, M., 2009. Engineering new paths to water data. Computers and Geosciences 35, 753–760.

Bih, J., 2006. Service Oriented Architecture – A new paradigm to implement dynamic E-business solutions, ubiquity, 07(30). http://ubiquity.acm.org/article.cfm?id=1159403.

Castronova, A.M., Goodall, J.L., Elag, M.M., 2013. Models as web services using the Open Geospatial Consortium (OGC) Web Processing Service (WPS) standard, Environmental Modelling & software, 41, 72-83.

Castronova, A.M., Goodall, J.L., Ercan, M.B., 2013. Integrated modeling within a hydro-logic information system: an OpenMI based approach, Environmental Modelling & software, 39, 263-273.

Chatterjee, S., 2004. Messaging Patterns in Service-Oriented Architecture,
http://msdn.microsoft.com/en-us/library/aa480027.aspx.

Chen, D., Shams, S., Carmona-Moreno, C., Leone, A., 2010. Assessment of Open Source GIS Software for Water Resources Management in Developing Countries, J of Hydro-Environment Research, 4 (3), 253-264.

Choi, K.W., Yu, K., 2008 Strategies for Introducing Geospatial One-Stop (GOS) to Fulfil Domestic Needs, International Journal of Geoinformatics, 4(1), 29-35.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S., 2002. Unraveling the Web Services Web, IEEE Internet Computing, 6, 86-93.

Facchi, A., Ortuani, B., Maggi, D., Gandolfi, C., 2004. Coupled SVAT – groundwater model for water resources simulation in irrigated alluvial plains, Environmental Modelling & Software, 19 (11), 1053–1063.

Feng, M., Liu, S., Euliss Jr., N.H., Young, C., Mushet, D.M., 2011. Prototyping an online wetland ecosystem services model using open model sharing standards. Environmental Modelling & Software, 26 (4), 458-468.

Foote, K.E., Lynch, M., 1995 Data Sources for GIS, Department of Geography, University of Texas at Austin.

Foster, I., 2005. Service-Oriented Science, Science, 308 (5723), 814-817.

Foster, I., Kesselman, C., NickJ.M., Tuecke, S., 2001. Grid services for distributed system integration, Computer, 35 (6), 37-46.

Georgakopoulos, D., Papazoglou, M., 2012. Service-Oriented Computing, Cambridge, Massachusetts, London, England.

Goodall, J.L., Horsburgh, J.S., Whiteaker, T.L. Maidment, D.R., Zaslavsky, I., 2008. A first approach to web services for the National Water Information System. Environmental Modeling & Software, 23 (4), 404-411.

Goodall, J.L., Robinson, B.F., Castronova, A.M., 2011. Modeling water resource systems using a service-oriented computing paradigm, Environmental Modelling & Software, 26(5), 573-582.

Goodchild, M.F., 2007. Citizens as sensors: the world of volunteered geography, GeoJournal 69 (4), 211–221.

Gornitz, V., 1995. Sea level rise: a review of recent past and near-future trends, Earth Surface Processes and Landforms, 20, 7-20.

Granell, C., Diaz, L., Gould, M., 2009. Distributed geospatial processing services, Encyclopedia of Information Science and Technology, Information Science Reference, 1186–1193

Granell, C., Diaz, L., Gould, M., 2010. Service-oriented applications for environ-mental models: reusable geospatial services, Environmental Modelling & Software, 25(2), 182-198.

Guariso, G., Hitz, M., Werthner, H., 1996. An integrated simulation and optimization modelling environment for decision support, Decision Support Systems, 16 (2), 103-117.

Hervouet, J.M., Bates, P., 2000. The TELEMAC modelling system Special issue, Hydrological Processes, 14(13), 2207-2208.

Horritt, M.S., Bates, P.D., 2001. Evaluation of 1D and 2D numerical models for predicting river flood inundation, Journal of Hydrology, 268 (1), 87-99.

Horsburgh, J.S., Tarboton, D.G., Maidment, D.R., Zaslavsky, I., 2011. Components of an environmental observatory information system. Computer & Geosciences, 37(2011), 207-218.

Horsburgh, J.S., Tarboton, D.G., Piasecki, M., Maidment, D.R., Zaslavsky, I., Valentine, D., Whitenack, T., 2009. An integrated system for publishing environmental observations data. Environmental Modelling & Software, 24 (8), 879-888.

House of Commons EFRA Committee, 2008. The potential of England's rural economy, Volume I, 5-9.

Huhns, M., Singh, M., 2005. Service-Oriented computing: key concepts and Principles, IEEE Internet Computing, 9(1), 75-81.

Jorgenson, M.T., Brown. J., 2005. Classification of the Alaskan Beaufort Sea Coast and estimation of carbon and sediments inputs from coastal erosion, Geo-Marine Letters, (25), 69-80.

Keller, M., Schimel, D.S., Hargrove, W.W., Hoffman, F.M. 2008. A Continental Strategy for the National Ecological Observatory Network, Frontiers in Ecology and the Environment, 6 (5), 282-284.

Khanbabaei, M., Asadi, M., 2011. Principles of Service-Oriented Architecture and Web Services Application In Order to Implement Service-Oriented Architecture in Software Engineering, Australian Journal of Basic and Applied Sciences, 5(11), 2046-2051.

Lecca, G., Petitdidier, M., Hluchy, L., Ivanovic, M., Kussul, N., Ray, N., Thieron, V., 2011. Grid computing technology for hydrological applications, Journal of Hydrology, 403(1-2), 186-199.

Leone, A., Chen, D., 2007. Implementation of an Object Oriented data model in an Information System for water catchment management: JAVA JDO and db4o Object Database. Environmental Modelling & Software, 22, 1805-1810.

Leone, A., Shams, S., Chen, D., 2006. An Object-Oriented and OpenGIS Supported Hydro Information System (30-HIS) for Upper Mersey River Basin Management. Journal of River Basin Management, 4(2), 1-9.

Liang, S.J., Molkenthin, F., 2001. A virtual GIS-based hydrodynamic model system for Tamshui River, Journal of Hydro informatics, 3(2011), 195-202.

Lutz, M., Sprado, J., Klien, E., Schubert, C., Christ, I., 2009. Overcoming semantic heterogeneity in spatial data infrastructures. Computers & Geosciences, 35(4), 739-752.

Maidment, D. R., 2008. Bringing Water Data Together, Journal of Water Resources Planning and Management, 134(2), 95-96.

Marks, K., Bates, P., 2000. Integration of high-resolution topographic data with floodplain flow models, Hydrological Processes, 14 (11-12), 2109-2122.

Maxwell, R.M., Miller, N.L., 2005. Development of a coupled land surface and groundwater model, Journal of Hydrometeorology, 6 (3), 233.

Miller, H. J., Han, J., 2009. Geographic data mining and knowledge discovery, Data mining and knowledge discovery series (2nd ed).

Mineter, M.J., Jarvis, C.H., Dowers, S., 2003. From stand-alone programs towards grid aware services and components: a case study in agricultural modelling with interpolated climate data. Environmental Modelling & Software, 18 (4), 379–391.

Morgan, J.E., McIntire, W.C., 1959. Quaternary geology of Bengal Basin, East Pakistan and India, Geological Society of America Bulletin, 70, 319-342.

OGC, 2014. Open Geospatial Consortium Making location count., http://www.opengeospatial.org/ogc.

Papajorgji, P., Beck, H.W., Braga, J.L., 2004. An architecture for developing service-oriented and component-based environmental models, Ecological Modelling, 179, 61-76.

Papazoglou, M.P., Georgakopoulos, D., 2003. Introduction: Service-oriented computing, Communications of the ACM - Service-oriented computing, 46(10), 24-28.

Paul, M., Ghosh, S.K., 2008. A framework for semantic interoperability for distributed geospatial repositories, Computing and Informatics, 27, 73-92.

Peckham, S. D., Goodall, J.L., 2013. Driving plug-and-play models with data from web services: A demonstration of interoperability between CSDMS and CUAHSI-HIS, Computers & Geosciences, 53, 154RLIN

Pingali, K., Stodghill, P., 2004. O'SOAP - A Web Services Framework for DDDAS Applications, Workshop on Dynamic Data-Driven Application Systems, International Conference on Computational Science, 3038, 797-804.

Pirie, D., 2012. Flood Warning Strategy 2012 to 2016, Scottish Environment Protection Agency, 5-18.

Pitt, M., 2008. An update of the Foresight Future Flooding 2004 qualitative risk analysis, Chapter 1, 3-4.

Robertson, G. P., L. W. Burger, C. L. Kling, R. Lowrance, and D. J. Mulla. 2007. New approaches to environmental management research at landscape and watershed scales, 27-50.

Richard, S.M., Clark, R., Grunberg, W., 2011. Application of the US Geoscience Information Network to deploying a National Geothermal Data System, Cambridge University, Cambridge, United Kingdom, 350-370.

Rizzoli, A., Donatelli, M., Athanasiadis, I., Villa, F., Huber, D., 2008. Semantic links in integrated modelling frameworks, Mathematics and Computers in Simulation, 78(2-3), 412-423.

Subramanya, K., 2008. Engineering Hydrology, McGraw-Hill, 75.

Shams, S., Huang. J., 2009. Object-Oriented Hydro Information System (OHIS) for the Estimation and Visualization of Vegetation Water Content, Journal of Engineering and Technology Vol. 7, No. 2 (JETIUT), 37- 50.

Shams, S., Zakzok, E., Chen, D., Huang, J., 2010. Estimation and monitoring non-point source pollutant loads: an object oriented hydro information approach, Int. J. Environment and Waste Management, 6(3/4), 220 - 236.

Singh, M.P., Huhns, M.N., 2005. Service-Oriented Computing: Semantics, Processes, Agents, 71-84.

Spanou, M., Chen, D., 2000. An Object-Oriented tool for the control of point-source pollution in river systems, Environmental Modelling and Software, 15(1), 35-54.

Sprott, D., Wilkes, L., 2004. Understanding Service-Oriented Architecture, Developer Network. http://msdn.microsoft.com/en-us/library/aa480021.aspx.

STOWA/RIZA, 1999. Smooth Modelling in Water Management, Good Modelling Practice Handbook, 30.

Tarboton, D.G., Horsburgh, J.S., Maidment, D.R., Whiteaker, T., Zaslavsky, I., Piasecki, M., Goodall, J., Valentine, D., Whitenack, T., 2009. Development of a community hydrologic information system. In: 18th World IMACS/MODSIM Congress. Cairns, Australia, 988-994.

Taylor, J., 2014. Implementation of NEON's quality assurance and quality control analyses and their quantification though a quality metric framework, Exhibit Hall, Sacramento Convention Center http://eco.confex.com/eco/2014/webprogram/Paper45307.html.

Telemac user manual, 2010. 2D hydrodynamics TELEMAC-2D software Version 6.0 USER MANUAL.

Thumerer, T., A. P. Jones, D. Brown., 2000. A GIS based coastal management system for climate change associated flood risk assessment on the east coast of England, International Journal of Geographical Information Science, 14(3), 265-281.

UN Population Division, 2010, World Population Prospects: The 2008 Revision and World Urbanization Prospects: The 2009 Revision, Retrieved on 18 October, 2010 from: http://esa.un.org/wup2009/unup/.

University of California San Diego, 2008. US Hydrologic information system initiative.

Verma,S., Verma, R.K., Singh, A., Naik, N.S., Web-Based GIS and Desktop Open Source GIS Software: An Emerging Innovative Approach for Water Resources Management, Advances in Computer Science,1061-1074.

Webber, J., Parastatidis, S., 2009. Realizing Service-Oriented Architectures with Web Services, 53-66.

WPS, Web Processing Service, OGC Making location count., http://www.opengeospatial.org/standards/wps.

Wyre Borough council, 2004. Wyre Flood and Coastal Defence Strategy Plan.

Zaslavsky, I., Maidment.,D. R., 2011. Service orientation in the design of a community hydrologic information system, 200.

Zaslavsky, I., D. Valentine, D. R. Maidment, 2007. CUAHSI cyberinfrastructure for hydrologic sciences, Geoinformatics, http://gsa.confex.com/gsa/2007GE/finalprogram/abstract_122270.htm.

Zaslavsky, I., D. Valentine, R. Hooper, M. Piasecki, A. Couch, A. Bedig, 2012, Community practices for naming and managing hydrologic variables, in Proceedings of the AWRA Spring Specialty Conference on GIS and Water Resources, New Orleans, LA, http://his.cuahsi.org/documents/IlyaZaslavsky_51e7c422_7956.pdf.

Zaslavsky, I., Valentine, D., Whiteaker, T., 2007. CUAHSI WaterML, OGC 07-041r1, Open Geospatial Consortium Discussion Paper, http://portal.opengeospatial.org/files/?artifact_id=21743.

Appendix I

SQLConnection.java

```
package lib;
import java.io.PrintStream;
import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
public class SQLConnection
{
   private static Connection db;
   private static Statement sql;
   private static DatabaseMetaData dbmd;
   private static ResultSet sqlresults;
   private static boolean isconnection=false;
   private static int totalnumbertable;
   private static int numberselcetNBV1;
   public static boolean isconnection(){
            return isconnection;
      }
   public static void connection()
   {
      String database = "//localhost:5432/mydb";
```

```
    String username = "postgres";
    String password = "zyc0107";
    try {
      Class.forName("org.postgresql.Driver");
      try {
        db = DriverManager.getConnection("jdbc:postgresql:" + database, username, password);
        dbmd = db.getMetaData();
        System.out.println("Connection to " + dbmd.getDatabaseProductName() + " " +
          dbmd.getDatabaseProductVersion() + " successful.\n");
        sql = db.createStatement();
        isconnection=true;
      }
      catch (SQLException e) {
        e.printStackTrace();
      }
    }
    catch (ClassNotFoundException e1) {
      e1.printStackTrace();
    }
}
public static Statement getsql()
{
    return sql;
}
public static ResultSet getsqlresults() {
    return sqlresults;
}
public static void selecttable(String tablename) {
    String sqlText = "select * from " + tablename;
    try {
      sqlresults = sql.executeQuery(sqlText);
    }
    catch (SQLException e)
    {
```

```
        e.printStackTrace();
      }
   }
   public static void inserttable(String nameofresulttable) {
         List<?> steplist = read_file_code.getsteplist();
         int maxnumberoftable=(int) (Math.ceil(steplist.size() / 150)+1);
         System.out.println(maxnumberoftable);
         List<Integer> fsearchpointgroup=read_fortran_files.getfsearchpointgroup();
         String   resulttablename= read_fortran_files.getresulttablename();
         System.out.println(fsearchpointgroup.get(1));
         System.out.println(resulttablename);
         for (int StepForRunPoint=0; StepForRunPoint<3;StepForRunPoint++){

               for (int i=1;i<=maxnumberoftable;i++){
                     String textString=null;
                     if(i==1){
                     textString = "SELECT * into " + nameofresulttable+StepForRunPoint+" FROM
"+resulttablename+"_"+i+"                                                                    WHERE
"+resulttablename+"_"+i+".gid="+fsearchpointgroup.get(StepForRunPoint);
                     }
                     else{

                     String textString_1 = " SELECT * FROM "+resulttablename+"_"+i+" WHERE
"+resulttablename+"_"+i+".gid="+fsearchpointgroup.get(StepForRunPoint);
                     textString="INSERT INTO "+nameofresulttable+StepForRunPoint+ textString_1;
                     }

                     System.out.print(textString);
         try
         {
            sql.executeUpdate(textString);
            System.out.println(textString);
         }
         catch (SQLException e) {
```

```
            e.printStackTrace();
            }
          }
          }

    }

    public static void creattable(String tablename) {
        String textString = "";
        String text = "";
        String textStringhead = "";
        String tablenametrim = tablename.trim();
        String NameofUnitTrim = tablename.trim();
        String NameofUnitTrimChange = "";
        if ((tablenametrim != null) || (tablenametrim != "")) {
            if (NameofUnitTrim.indexOf(" ") != -1)
                NameofUnitTrimChange = NameofUnitTrim.replace(" ", "_");
            else {
                NameofUnitTrimChange = NameofUnitTrim;
            }

            List<?> step = read_file_code.getsteplist();
            List<?> AT = read_file_code.gettimelist();
            textStringhead = "Number varchar(40), X double precision,Y double precision,";
            for (int i = 0; i < step.size() - 1; i++) {
                int timeInt = ((Float)AT.get(i)).intValue();
                String time = Integer.toString(timeInt).trim();
                text = NameofUnitTrimChange + time + " varchar(40),";
                textString = textString + text;
            }
            String lasttime = Integer.toString(((Float)AT.get(AT.size() - 1)).intValue()).trim();
            String string = NameofUnitTrimChange + lasttime + " varchar(40)";
            String sqlText = "create table IF NOT EXISTS " + NameofUnitTrimChange + "(" + textStringhead +
textString + string + ")";
```

```
      System.out.print(sqlText);
      try {
        sql.executeUpdate(sqlText);
      }
      catch (SQLException e) {
        e.printStackTrace();
      }
    }
}
public static void updatetable(String tablename, String NameofUnit,int tableofnumber)
{
   String sqlText = "";
   ArrayList<String> arr = new ArrayList<String>();
   String NameofUnitTrim = NameofUnit.trim();
   String NameofUnitTrimChange = "";
   int NPOIN = read_file_code.getNPOIN();
   if ((NameofUnitTrim != null) || (NameofUnitTrim != "")) {
      if (NameofUnitTrim.indexOf(" ") != -1)
        NameofUnitTrimChange = NameofUnitTrim.replace(" ", "_");
      else {
        NameofUnitTrimChange = NameofUnitTrim;
      }
      List<?> steplist = read_file_code.getsteplist();
      List<?> AT = read_file_code.gettimelist();
      Object[][][] totalvariable = read_file_code.gettotalvariable();
      numberselcetNBV1=read_fortran_files.getnumberselcetNBV1();
      String textString = "";
      String text = "";
      String string = "";
      totalnumbertable = (int)Math.ceil(steplist.size() / 150)+1;
      System.out.println("totalnumbertable"+totalnumbertable);
        for (int n = 0; n < NPOIN; n++)
        {
          if (tableofnumber == totalnumbertable) {
```

```
                System.out.println("totalnumbertable1"+totalnumbertable);
                tableofnumber=tableofnumber-1;
                 for (int i = tableofnumber * 150; i < steplist.size(); i++) {
                    int subStep=i%150;
                    int timeInt = ((Float)AT.get(i)).intValue();
                    String time = Integer.toString(timeInt).trim();
                    if (((i + 1) % 150 != 0) && (i != steplist.size() - 1)) {
                      text = NameofUnitTrimChange + time + " =" +
totalvariable[numberselcetNBV1][n][subStep] + ",";
                      textString = textString + text;
                    } else {
                      String lasttime = Integer.toString(((Float)AT.get(i)).intValue()).trim();
                      string = NameofUnitTrimChange + lasttime + " =" +
totalvariable[numberselcetNBV1][n][subStep];
                      int a = n + 1;
                      tableofnumber = tableofnumber + 1;
                      sqlText = "UPDATE " + tablename + "_" + tableofnumber + " SET " + textString +
string + " where gid=" + a;
                      arr.add(sqlText);
                      textString = "";
                      text = "";
                      string = "";
                      System.out.println("totalnumbertable"+sqlText);
                    }
                  }
                }
                else {
                  tableofnumber=tableofnumber-1;
                   for (int i = 0; i < 150; i++) {
                    int timeInt = ((Float)AT.get(i + tableofnumber * 150)).intValue();
                    String time = Integer.toString(timeInt).trim();
                    if (((i + 1) % 150 != 0) && (i != steplist.size() - 1)) {
                      text = NameofUnitTrimChange + time + " =" + totalvariable[numberselcetNBV1][n][i]
+ ",";
```

```
                    textString = textString + text;
                } else {
                    String lasttime = Integer.toString(((Float)AT.get(i + tableofnumber * 150)).intValue()).trim();
                    string = NameofUnitTrimChange + lasttime + " =" + totalvariable[numberselcetNBV1][n][i];
                    int a = n + 1;
                    tableofnumber = tableofnumber + 1;
                    sqlText = "UPDATE " + tablename + "_" + tableofnumber + " SET " + textString + string + " where gid=" + a;
                    arr.add(sqlText);

                    textString = "";
                    text = "";
                    string = "";
                }
              }
            }

            if ((n % 500 == 0 && n!=0) || (NPOIN - 1 - n <= 150)){
            try {
              for (int i = 0; i < arr.size(); i++) {
                sql.addBatch((String)arr.get(i));
                if (i==arr.size()-1 && tableofnumber==8){
                System.out.println("Executing this command: " + (String)arr.get(i) + "\n");
                }
               }
              sql.executeBatch();
              arr.clear();
              System.out.println("n " + n);
            }
            catch (SQLException e)
            {
              e.printStackTrace();
```

```
            }
          }
          }

          totalvariable=null;
    }
}
public static void droptable(String tablename)
{
    String sqlText = "";
    String tablenametrim = tablename.trim();
    String NameofUnitTrim = tablename.trim();
    String NameofUnitTrimChange = "";
    if ((tablenametrim != null) || (tablenametrim != "")) {
      if (NameofUnitTrim.indexOf(" ") != -1)
        NameofUnitTrimChange = NameofUnitTrim.replace(" ", "_");
      else {
        NameofUnitTrimChange = NameofUnitTrim;
      }
      sqlText = "DROP table Export_" + NameofUnitTrimChange;
      try
      {
        sql.executeUpdate(sqlText);
        System.out.print("OK");
      }
      catch (SQLException e) {
        e.printStackTrace();
      }
    }
}
public static void altertable(String tablename, String NameofUnit,String serialnum )
{
    String text = "";
    String sqlText = "";
```

```
String NameofUnitTrim = NameofUnit.trim();
String NameofUnitTrimChange = "";
int numbertable = 0;
boolean go = true;
if ((NameofUnitTrim != null) || (NameofUnitTrim != "")) {
  if (NameofUnitTrim.indexOf(" ") != -1)
    NameofUnitTrimChange = NameofUnitTrim.replace(" ", "_");
  else {
    NameofUnitTrimChange = NameofUnitTrim;
  }
  List<?> step = read_file_code.getsteplist();
  List<?> AT = read_file_code.gettimelist();
  for (int i = 0; i < step.size(); i++)
  {
    if (((i % 150 == 0) || (step.size() - 1 - (numbertable - 1) * 150 < 150)) && go==true)
    {
        numbertable++;
        System.out.println(i);
        sqlText = "select * into " + tablename+serialnum + "_" + numbertable + " from " +
tablename;
        if (step.size() - 1 - i < 150){
          go = false;
        }
      try
      {
        sql.executeUpdate(sqlText);
        System.out.println(sqlText);
      }
      catch (SQLException e) {
        e.printStackTrace();
      }
    }
    System.out.println(i);
    int timeInt = ((Float)AT.get(i)).intValue();
```

```
            String time = Integer.toString(timeInt).trim();
            text = NameofUnitTrimChange + time + " numeric";
            sqlText = "alter table " + tablename+serialnum    + "_" + numbertable + " add " + text;
            try
            {
               sql.executeUpdate(sqlText);
               System.out.println(sqlText);
            }
            catch (SQLException e) {
               e.printStackTrace();
            }
         }
         }
   }
   public static void closed()
   {
      try
      {
         db.close();
      }
      catch (SQLException e) {
         e.printStackTrace();
      }
   }
}
```

OpenFile.java

```
package lib;
import java.io.File;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileFilter;
public class OpenFile {
	static String path;
	 public static void openfile() {
		int flag;
		 JFileChooser fileChooser = new JFileChooser();
		 fileChooser.setCurrentDirectory(new File("."));
		 fileChooser.setAcceptAllFileFilterUsed(false);
		 final String[][] fileENames = { { ".java", "JAVA 源程序  文件(*.java)" },
				{ ".res", "Telemac  文件(*.res)" },
				{ ".xls", "MS-Excel 2003  文件(*.xls)" }
				 };
		 fileChooser.addChoosableFileFilter(new FileFilter() {
			public boolean accept(File file) {
			 return true;
			}
			public String getDescription() {
			 return "所有文件(*.*)";
			}
		       });
		 for (final String[] fileEName : fileENames) {
		  fileChooser.setFileFilter(new javax.swing.filechooser.FileFilter() {
		    public boolean accept(File file) {
		     if (file.getName().endsWith(fileEName[0]) || file.isDirectory()) {
		      return true;
		     }
		     return false;
		    }
```

```
                public String getDescription() {
                  return fileEName[1];
                 }
                });
              }
        flag=fileChooser.showDialog(null, null);
                  if(flag==JFileChooser.APPROVE_OPTION){
                  String path=fileChooser.getSelectedFile().getPath();
                  OpenFile.path=path;
             }
       }
      public static String getrespath() {
                return path;
               }
}
```

Read_file_code.java

```
package lib;
import java.awt.Label;
import java.io.BufferedInputStream;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
public class read_file_code {
	static DataInputStream dis;
	static int Take_Number;
//	static Object[][] variable;//变量，NPOIN
	static Object[][][] totalvariable;//变量，NPOIN
	static int NBV1;
	static int NBV2;
	static int NPOIN;
	static int NELEM;
	static float AT;
	static List<Float> timelist;
	static List<Integer> steplist;
	static String[] NameofUnit;
	static String[] Unit;
	static Object[] X;
	static Object[] Y;
	static byte[] itemBuf;
	static String charTake_Number;
	static boolean initial=true;
	static boolean sencenumber=false;
```

```
	public static boolean sence(int Take_Number){
		boolean sence=false;
		if (Take_Number>0&Take_Number<1000){
			sence=true;
		}else sence=false;
		return sence;
	}
	public static int getNBV1(){
		return NBV1;
}
	public static DataInputStream getdis(String fileName){
			return dis;
	}
	public static String[] getNameofUnit(){
		return NameofUnit;
	}
	public static Object[][][] gettotalvariable(){
		return totalvariable;
	}
	public static int getvariablenumber(){
		return NBV1;
	}
	public static int getNPOIN(){
		return NPOIN;
	}
	public static int getNELEM(){
		return NELEM;
	}
	public static Object[] getX(){
		return X;
	}
	public static Object[] getY(){
		return Y;
	}
```

```
 public static List<Integer> getsteplist(){
        return steplist;
 }
 public static List<Float> gettimelist(){
        return timelist;
 }
public static void readFile(String fileName,int filenumber){
 read_file_code.totalvariable=null;
      File file = new File(fileName);
      try {
        DataInputStream dis = new DataInputStream(
                        new BufferedInputStream(
                            new FileInputStream(file)));
                    read_file_code.dis=dis;
                    System.out.println("开始读取文件");
                    byte[] itemBuf = new byte[80];
                    read_file_code.itemBuf=itemBuf;
                    //80
                    boolean logic=false;
                    while (logic==false) {
                          int t = dis.readInt();
                          logic=sence(t);
                          Take_Number=t;
                    }
                    dis.read(itemBuf, 0, 72);
                    String title   = new String(itemBuf,0,72);
                     System.out.println("title:"+ title);
                     dis.read(itemBuf, 0, 8);
                     String format   = new String(itemBuf,0,8);
                     System.out.println("format:"+ format);
                    //80
                    logic=false;
                    while (logic==false) {
                          int t = dis.readInt();
```

```
                logic=sence(t);
                Take_Number=t;
            }
            dis.readInt();//8
        int NBV1 = dis.readInt(); //number of variable     //9
        System.out.println("Number of NBV1 "+NBV1);
        int NBV2 = dis.readInt();//0
        System.out.println("Number of NBV2 "+NBV2);
      String[] NameofUnit = new String[NBV1];
      String[] Unit = new String[NBV1];
        dis.readInt(); //8
        int unit = dis.readInt();//32
        for(int i=0; i <NBV1; i++){
        int halfunit= unit/2;
          dis.read(itemBuf, 0, halfunit);
          String variable_name    = new String(itemBuf,0,halfunit);
          NameofUnit[i]=variable_name;
          dis.read(itemBuf, 0, halfunit);
          String variable_name_units    = new String(itemBuf,0,halfunit);
          Unit[i]=variable_name_units;
          System.out.println("变量名称:"+ NameofUnit[i]+Unit[i]);
          dis.read(itemBuf, 0, 8);
        }
        System.out.println("NBV1 变量个数:"+ NBV1);
        System.out.println("NBV2 变量个数:"+ NBV2);
        //      dis.readInt();//32
        //      dis.readInt();//40
//IPARAM
        boolean gomore = false;
        for (int i=0; i<10; i++){
                if (i==9){
                int panduan=dis.readInt();

                if(panduan==1){
```

```
                gomore=true;
                }
                else {
                gomore=false;
                }
    }else {
          dis.readInt();
          }
}

int FORMAT = dis.readInt();
int NELEM=0;
int NPOIN=0;
int NDP=0;
if (FORMAT==40){//SERAPHIN FORMAT FILE
       if (gomore==true){
            for (int i=0; i<8; i++){
                 dis.readInt();
            }
        }
    dis.readInt();//16
    NELEM = dis.readInt();//number of elements
    NPOIN = dis.readInt();//number of points
    NDP = dis.readInt();// number of points per element
    dis.readInt();//1

    System.out.println("NELEM "+NELEM);
    System.out.println("NPOIN "+NPOIN);
    System.out.println("NDP "+NDP);
}

//IKLE
dis.readInt();//16
```

```
                dis.readInt();//196584

                Object[][] NELEMList = new Object[NELEM][NDP];
                int i=0;
                Object[] NELEMEach;
                while (i<NELEM){
                    NELEMEach = new Object[NDP];
                    for (int n=0; n<NDP;n++){
                    NELEMEach[n] = (Object) dis.readInt();
                    }
                    if (i==0||i==NELEM-1){//last 2628 3374 5797 node
                    System.out.println("NELEMList[i++]     "+NELEMEach[0]+"     "+NELEMEach[1]+"
"+NELEMEach[2]);
                    }

                    NELEMList[i] = NELEMEach;
                    i=i+1;

                }
                //IPOBO
                dis.readInt();//196584
                dis.readInt();//33940
                Object[] IPOBO = new Object[NPOIN];
                for (int n=0; n<NPOIN;n++){
                    IPOBO[n] = (Object) dis.readInt();
                }

                //x
                dis.readInt();//33940
                dis.readInt();//33940

                Object[] X = new Object[NPOIN];
                for (int n=0; n<NPOIN;n++){
                    X[n] = (Object) dis.readFloat();
```

```
        }
//          System.out.println("X[n] "+X[2]);

      //Y
        dis.readInt();//33940
        dis.readInt();//33940

        Object[] Y = new Object[NPOIN];
        for (int n=0; n<NPOIN;n++){
            Y[n] = (Object) dis.readFloat();
        }
        int Step = 0;

        boolean go = true;
      List<Float> timelist = new ArrayList<Float>();
      List<Integer> steplist = new ArrayList<Integer>();
      Object[][][] totalvariable= new Object[NBV1][NPOIN][150];
        while(go==true) { //第一次循环，为的是找到 step 值
            try{

            //    System.out.println("第"+Step+"输出开始");
                int subStep=Step%150;
                int numberoftable=(int)Math.floor(Step / 150)+1;
                //每次 step ，是循环 NBV1 次，录入 NBV1 个变量
                for (i=0; i<NBV1;i++){
                    //定义 NBV1 个循环开始前的头
                int number=0;
                if (i==0){
                    while(number<5) {
                            if (number!=2){
                                dis.readInt();
                            }else {

                                Float AT=dis.readFloat();
```

```
                        timelist.add(AT);
                    }
                    number=number+1;
                }
            }else
                while(number<2) {
                    dis.readInt();
                    number=number+1;
                }

            number=0;
                // 头之后 又录入了 NPOIN 个变量
                while(number<NPOIN) {

                    if (numberoftable==filenumber){
                    totalvariable[i][number][subStep]=(Object) dis.readFloat() ;
                    }else {
                        dis.readFloat();
                    }

                    number=number+1;
                }

            }
            steplist.add(Step);
            System.out.println("第"+Step+"步");
            if (numberoftable==filenumber){
                System.out.println("第"+timelist.get(Step)+"秒的数值");
                for (int n=0;n<NBV1;n++){
                    System.out.println(NameofUnit[n]+"    "+Unit[n]+"    "+NPOIN+"
"+totalvariable[n][NPOIN-1][subStep]);
                }
                System.out.println("第"+Step+"输出完毕");
            }
```

```
                read_file_code.dis=dis;
                if (filenumber==1){
                read_file_code.dis=dis;
                read_file_code.NBV1= NBV1;
                read_file_code.NBV2= NBV2;
                read_file_code.NPOIN= NPOIN;
                read_file_code.NELEM= NELEM;
                read_file_code.timelist= timelist;
                read_file_code.steplist= steplist;
                read_file_code.NameofUnit= NameofUnit;
                read_file_code.Unit= Unit;
                read_file_code.X= X;
                read_file_code.Y= Y;
                read_file_code.itemBuf= itemBuf;
                }
                read_file_code.totalvariable=totalvariable;
                      Step=Step+1;
                }catch(java.io.EOFException e){
                      go=false;
                }

            }
            totalvariable=null;
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
      }
}
```

Read_fortran_files.java

```
package lib;
import java.awt.Choice;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.io.DataOutputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.List;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.SpringLayout;
import javax.swing.border.EmptyBorder;
import lib.OpenFile;
import lib.SQLConnection;
import lib.read_file_code;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JTextField;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
public class read_fortran_files extends JFrame
{
   static Object[][] variable;
```

```
    static int NBV1;
    static int NBV2;
    static int NPOIN;
    static int NELEM;
    static String resulttablename;
    static String[] NameofUnit;
    static String[] Unit;
    static Object[] X;
    static Object[] Y;
    private JPanel contentPane;
    private Choice choice;
    private JTextField txtResulttable;
    private JTextField textField;
    private JTextField textField_1;
    private JTextField textField_2;
    private JTextField txtBlackpool;
    private JComboBox<String> choice_1=new JComboBox<String>();
    static String fsearchresulttablename;

    private static List<Integer> fsearchpointgroup;
    static int numberselcetNBV1;
    private JTextField txtA;

        public static String getfsearchresulttablename(){
            return fsearchresulttablename;
}

        public static List<Integer> getfsearchpointgroup(){
            return fsearchpointgroup;
}

        public static int getnumberselcetNBV1(){
            return numberselcetNBV1;
}
```

```
    public static String getresulttablename(){
            return resulttablename;
}

  public static void main(String[] args)
  {
    EventQueue.invokeLater(new Runnable() {
      public void run() {
        try {
          read_fortran_files frame = new read_fortran_files();
          frame.setVisible(true);
        } catch (Exception e) {
          e.printStackTrace();
        }
      }
    });
  }
  public read_fortran_files()
  {
     txtBlackpool = new JTextField();
     txtBlackpool.setText("blackpool");

    choice_1.setBounds(423, 179, 76, 21);
    final JButton btnImport = new JButton("Import");
    btnImport.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
     }
    });
    btnImport.setBounds(171, 241, 101, 23);
    final JButton btnAlterTable = new JButton("Alter Table");
    btnAlterTable.setBounds(45, 241, 108, 23);
    btnAlterTable.setEnabled(false);
    setDefaultCloseOperation(3);
    setBounds(150, 150, 532, 378);
```

```
this.contentPane = new JPanel();
this.contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(this.contentPane);
JButton btnReadfortranfiles = new JButton("OK");
btnReadfortranfiles.addActionListener(new ActionListener() {
 public void actionPerformed(ActionEvent e) {
 }
});
btnReadfortranfiles.setBounds(434, 110, 65, 23);
btnReadfortranfiles.addMouseListener(new MouseAdapter(){
 public void mouseClicked(MouseEvent arg0) {
      System.out.println("开始");
      SQLConnection.connection();
      String filepath = read_fortran_files.this.choice.getSelectedItem();
      read_file_code.readFile(filepath,1);
      boolean connectionsuccessful = SQLConnection.isconnection();
      NameofUnit = read_file_code.getNameofUnit();
      NBV1=read_file_code.getNBV1();
      if (connectionsuccessful) {
         btnImport.setEnabled(true);
         btnAlterTable.setEnabled(true);
         NBV1=read_file_code.getNBV1();

         choice_1.removeAllItems();
         for (int i=0;i<NBV1;i++){
           choice_1.addItem(NameofUnit[i]);
         }
      }
      System.out.println("结束");
   }
});
JButton btnNewButton = new JButton("Exit");
btnNewButton.setBounds(400, 241, 99, 23);
btnNewButton.addMouseListener(new MouseAdapter()
```

```
{
   public void mouseClicked(MouseEvent arg0) {
      System.exit(0);
   }
});
contentPane.setLayout(null);
this.contentPane.add(btnReadfortranfiles);
JLabel lblNewLabel = new JLabel("Open Fortran Files");
lblNewLabel.setBounds(182, 15, 246, 100);
lblNewLabel.setFont(new Font("宋体", 0, 15));
this.contentPane.add(lblNewLabel);
this.contentPane.add(btnNewButton);
this.choice = new Choice();
choice.setBounds(192, 111, 150, 21);
String filepath1 = "./result/lbay.res";
String filepath2 = "F:/算例 blackpool 数据/算例/模拟数据/res/T2DRES_success_13";
String filepath3 = "./result/LbayTel.res";
this.choice.addItem(filepath1);
this.choice.addItem(filepath2);
this.choice.addItem(filepath3);
this.contentPane.add(this.choice);

JButton btnNewButton_1 = new JButton("Open");
btnNewButton_1.setBounds(352, 110, 76, 23);
btnNewButton_1.addActionListener(new ActionListener() {
 public void actionPerformed(ActionEvent e) {
 }
});
btnNewButton_1.addMouseListener(new MouseAdapter()
{
   public void mouseClicked(MouseEvent e) {
      OpenFile.openfile();
      String path = OpenFile.getrespath();
      read_fortran_files.this.choice.addItem(path);
```

```
      read_fortran_files.this.choice.select(path);
    }
});
this.contentPane.add(btnNewButton_1);
btnImport.setEnabled(false);
btnImport.addMouseListener(new MouseAdapter()
{
   public void mouseClicked(MouseEvent arg0) {
      if (btnImport.isEnabled())
      {
         List<?> steplist = read_file_code.getsteplist();
         int totalnumbertable = (int)Math.ceil(steplist.size() / 150)+1;
         numberselcetNBV1=choice_1.getSelectedIndex();
         for (int i=1;i<=totalnumbertable;i++){
           String filepath = read_fortran_files.this.choice.getSelectedItem();
           System.out.println("(filepath,i)"+i);
           read_file_code.readFile(filepath,i);
           resulttablename=txtBlackpool.getText()+txtA.getText();

           SQLConnection.updatetable(resulttablename, choice_1.getSelectedItem().toString(),i);
         }
         SQLConnection.closed();
      }
   }
});
this.contentPane.add(btnImport);
JButton btnExportXy = new JButton("Export X,Y");
btnExportXy.addActionListener(new ActionListener() {
 public void actionPerformed(ActionEvent e) {
 }
});
btnExportXy.setBounds(282, 241, 108, 23);
btnExportXy.addMouseListener(new MouseAdapter()
{
```

```
public void mouseClicked(MouseEvent arg0) {
   FileOutputStream f_write = null;
   try
   {

   String path="d:/"+txtBlackpool.getText()+".txt";
      f_write = new FileOutputStream(path);
   }
   catch (FileNotFoundException e1) {
      e1.printStackTrace();
   }
   DataOutputStream dos = new DataOutputStream(f_write);
   System.out.println("Input:");
   Object[] X = read_file_code.getX();
   Object[] Y = read_file_code.getY();
   int NPOIN = read_file_code.getNPOIN();
   try {
      dos.writeBytes("X,Y");
      dos.writeBytes("\r\n");
   }
   catch (IOException e1) {
      e1.printStackTrace();
   }
   for (int n = 0; n < NPOIN; n++)
   {
      String XString = X[n].toString();
      String YString = Y[n].toString();
      String xyString = XString + ",\t" + YString;
      try
      {
         dos.writeBytes(xyString);
         dos.writeBytes("\r\n");
         System.out.print(xyString);
      }
```

```
            catch (Exception e) {
               throw new InternalError("Unexpected CloneNotSUpportedException: " + e.getMessage());
            }
         }
         try
         {
            dos.close();
         }
         catch (IOException e) {
            e.printStackTrace();
            btnAlterTable.setEnabled(true);
         }
      }
   });
   this.contentPane.add(btnExportXy);
   txtResulttable = new JTextField();
   txtResulttable.setBounds(171, 210, 171, 21);
   txtResulttable.setText("resulttable");

   btnAlterTable.addMouseListener(new MouseAdapter()
   {
      public void mouseClicked(MouseEvent arg0) {
       SQLConnection.altertable(txtBlackpool.getText(),
choice_1.getSelectedItem().toString(),txtA.getText());
      }
   });
   this.contentPane.add(btnAlterTable);

   JButton btnNewButton_2 = new JButton("Result");
   btnNewButton_2.setBounds(370, 209, 129, 23);
   btnNewButton_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
    }
   });
```

```
btnNewButton_2.addMouseListener(new MouseAdapter() {
 @Override
 public void mouseClicked(MouseEvent e) {
         System.out.print("textString");
         fsearchpointgroup=null;
         resulttablename=txtBlackpool.getText()+txtA.getText();
            List<Integer> searchpointgroup = new ArrayList<Integer>();
             int searchpointgroup_1=Integer.parseInt(textField.getText());
             int searchpointgroup_2=Integer.parseInt(textField_1.getText());
             int searchpointgroup_3=Integer.parseInt(textField_2.getText());

             searchpointgroup.add(searchpointgroup_1);
             searchpointgroup.add(searchpointgroup_2);
             searchpointgroup.add(searchpointgroup_3);

             fsearchpointgroup=searchpointgroup;

         fsearchresulttablename=txtResulttable.getText()+txtA.getText();
        SQLConnection.inserttable(fsearchresulttablename);

 }
});
contentPane.add(btnNewButton_2);
contentPane.add(txtResulttable);
txtResulttable.setColumns(10);

JLabel lblText = new JLabel("Creat table name:");
lblText.setBounds(45, 151, 134, 15);
contentPane.add(lblText);

JLabel lblChooseTheResult = new JLabel("Choose the result files:");
lblChooseTheResult.setBounds(45, 108, 176, 27);
contentPane.add(lblChooseTheResult);
```

```
        JLabel lblChooseThe = new JLabel("Choose the Row:");
        lblChooseThe.setBounds(45, 182, 99, 15);
        contentPane.add(lblChooseThe);

        textField = new JTextField();
        textField.setText("13880");
        textField.setBounds(171, 179, 76, 21);
        contentPane.add(textField);
        textField.setColumns(10);

        textField_1 = new JTextField();
        textField_1.setText("18840");
        textField_1.setBounds(257, 179, 76, 21);
        textField_1.setColumns(10);
        contentPane.add(textField_1);

        textField_2 = new JTextField();
        textField_2.setText("16400");
        textField_2.setBounds(343, 179, 76, 21);
        textField_2.setColumns(10);
        contentPane.add(textField_2);

        JLabel lblResultTableName = new JLabel("Result table name:");
        lblResultTableName.setBounds(45, 207, 108, 15);
        contentPane.add(lblResultTableName);



//      textField_4 = new JTextField();
        txtBlackpool.addMouseListener(new MouseAdapter() {
         @Override
         public void mouseClicked(MouseEvent e) {
         }
        });
```

```
        txtBlackpool.setBounds(170, 148, 76, 21);
        txtBlackpool.setColumns(10);
        contentPane.add(txtBlackpool);
        contentPane.add(choice_1);

        JLabel lblSer = new JLabel("Serial Number:");
        lblSer.setBounds(260, 151, 99, 15);
        contentPane.add(lblSer);

        txtA = new JTextField();
        txtA.setText("a");
        txtA.setColumns(10);
        txtA.setBounds(352, 148, 147, 21);
        contentPane.add(txtA);
    }
private List<Integer> VaulttoInt(String text) {
        // TODO Auto-generated method stub
        return null;
}
}
```