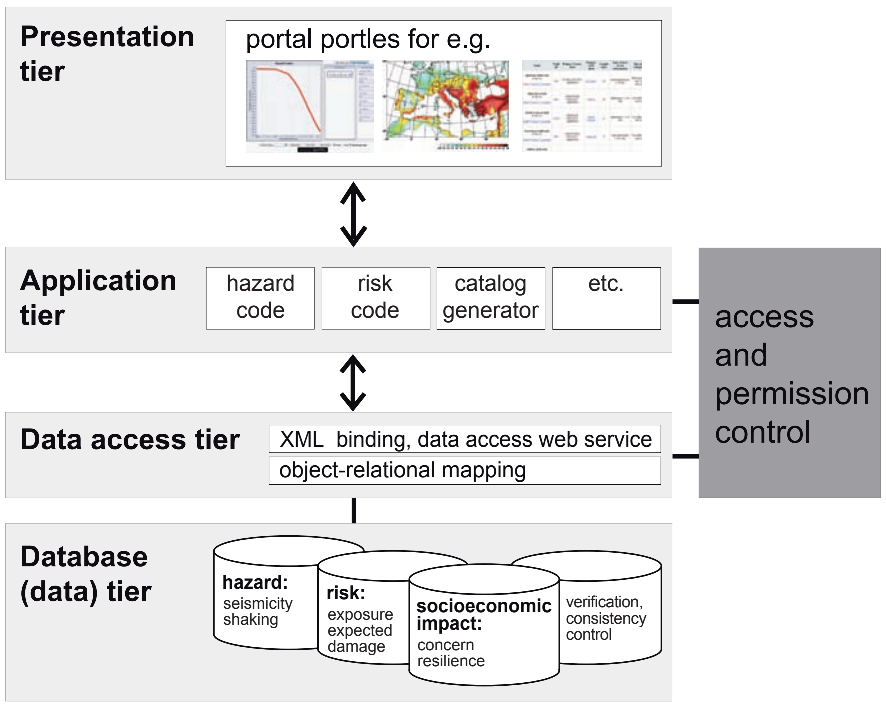
GEM Technical Report 2010-6

OpenGEM System Design Document

R. Krishnamurthy, F. Euchner, A. Mömke, S. Roland, P. Kästli


Presentation tier
portal portles for e.g.
Application tier
hazard code
risk code
catalog generator
etc.
Data access tier
XML binding, data access web service
object-relational mapping
Database (data) tier
hazard: seismicity shaking
risk: exposure expected damage
socioeconomic impact: concern resilience
verification, consistency control
access and permission control

OpenGEM System Design Document

By R. Krishnamurthy[1], F. Euchner[1], A. Mömke[1], R. Siegert[1], P. Kästli[1]

October 2010

1. *Swiss Seismological Service, ETH Zurich, Zurich, Switzerland*

Citation: Krishnamurthy R., Euchner F., Mömke A., Siegert R., Kästli P. (2010) OpenGEM System Design Document, GEM Technical Report 2010-6, GEM Foundation, Pavia, Italy.

www.globalquakemodel.org

ABSTRACT

The aim of this document is to provide an overview of the design of the OpenGEM system, within the scope of the GEM1 project. OpenGEM is the name that was chosen for the IT platform which will allow calculation and communication of earthquake risk on a global scale. One of the goals of the GEM1 project was to design an initial model building structure and this report hence describes the various components of the design. The IT infrastructure described in this report has been reviewed during an intensive IT-review after which a strategy has been determined for further development of GEMs IT architecture, which makes use of the work done during GEM1, but has taken a different approach. For more information on the IT review and a summary of the report, please visit: http://www.globalquakemodel.org/node/928.

GLOSSARY

Term	Definition
SDD	Software Design Document
SOA	Service Oriented Architecture
ORM	Object Relation Mapping
UI	User Interface
GUI	Graphical User Interface
VM	Virtual Machine
GA	General Availability
RC	Release Candidate
FOSS	Free Open Source Software
OASIS	Organization for the Advancement of Structured Information Standards
WMS	Web Map Service, an OGC standard to provide localized imaginary for mapping purposes over a web service, along with specified metadata
WFS	Web Feature Service
KML	“Keyhole Markup Language”, an XML dialect introduced by Google to describe vectorbased and rasterbased geodata. Compared to GML, KML concentrates on visualization, while GML allows for generic object properties.
GML	“Geography Markup Language”. An XML dialect of the Open Geospatial Consortium and ISO to describe spatial data along with sensor- or measurement data. GML is used e.g. for WFS payloads.
WEB Service	A web based data or functionality service identified by an uniform resource identifier (URI), and described in its capabilities by an XML (WDSL) document.
Portlet	A portlet is a software component that produces a part of a user interface e.g. as html code, for display by a portal server in a portal. Portlets following the JSR 168 and JSR 168 standards are deployable in most existing portal servers, and allow standardized inter-portlet communication (JSR 268)
OGC	“Open Geospatial Consortium”, an NGO invested in standardization and interoperability of spatial information and related service
JSR 168/268	Java specification requests (de-facto standard documents) for portlets. JSR 168 defines the Portlet API 1.0, handling mostly the Portal-Portlet interaction; JSR 286 defines the Portlet API 2.0, which has notable extensions especially for inter-portlet communication
SOAP	Simple Object Access Protocol, a protocol for allowing session-based communication of function calls and data between remote systems and applications, e.g. for use with web services. Soap bases on xml-messages, and is independent of operating systems and code languages.
REST	Representational State Transfer, another standard for web service interactions. REST bases on HTTP get, put, post, and deletes requests. It is per se stateless.

TABLE OF CONTENTS

1 Introduction

The GEM project plans to compile its earthquake hazard and risk results as a *living model*. An IT-platform, called OpenGEM, should integrate base data, the model definitions and their variants, the codes for assessing hazard, risk, and socio-economic impact (along with computing resources to run them), and their results in a simple and functional form. All aspects of the system should be accessible through a web-mounted portal. All software is intended to be developed as open source and following open standards, for code as well as for data formats and service interfaces. The user interface is to be provided as a portal/portlet environment allowing and adequate to be integrated with other IT initiatives within, and beyond, the seismic community. Within the GEM1 project, an prototype was developed to serve as a proof-of-concept for the intended IT infrastructure to host the living model.

1.1 Purpose and Scope

This document provides the detailed design of *OpenGEM*, as was developed on a proof-of-concept level within GEM1. It describes both general concepts and implemented components, and serves as a basis for an architectural review before upgrading it to an operational and full-featured living model environment.

1.2 Availability

As the frontends that were developed within GEM1, as well as the data transported by them, have a preliminary character, they are not open to the public. The testbed installation is available at http://opengem.ethz.ch:8080. It is password protected.

1.3 Outline of the Report

After the Introduction section, the outline of the report is as follows: Section 2 deals with all the Design Considerations, with sub-sections – Assumptions and Dependencies, General Constraints, Goals and Guidelines, Development Methods and Architectural Strategies. Section 3 regards the Architectural Design that specifies the design entities that collaborate to perform all the functions included in the system. Each of these entities has an Abstract description concerning the services that it provides to the rest of the system. Section 4 is concerns with the various policies governing the development and use of the *OpenGEM* system. Section 5 contains the Use Case descriptions. Each Use Case stated can be traced by the given design, Input/Output UI and Sequence of Interaction. Section 6 covers the detailed System Design.

2 Design Considerations

2.1 Assumptions and Dependencies

There has been no formal requirements-gathering effort, leading to a Requirements Definition Document. The requirements have been collected by informal discussion with the stakeholders, sponsors and representatives of Regional Programmes such as (SHARE (Seismic Hazard Harmonization in Europe) and EMME (Earthquake Model of the Middle East)). Hence, this document describes the *OpenGEM system* only as an iterative prototype. The document facilitates further discussion, as the system needs further review and input on final requirements.

A community of prospective end-users of the *OpenGEM* system is growing. Even if currently the focus will be first on internal users and stakeholders directly involved in GEM, it is highly likely that there will be new user requirements which were not anticipated.

2.2 General Constraints

#	Development Constraints	Remarks
1	End-user environment	OpenGEM is currently designed for a WebGUI based access to server-provided distributed services. Further options, such as a command line interface or standalone desktop software for sub-aspects, are not pursued at this moment.
2	Standards Compliance – Operating System	The *OpenGEM* frontend is browser mounted and thus system independent. Software development for the core server components is done in Java and around PostgreSQL. These are open software packages which are available for many operating systems. However, core development work is done on linux (notably scientific in linux 5.3); no further compatibility testing is done for server-side components.
3	Standards Compliance – Development Environment	Use of Java (JEE 1.6 or later) and related technologies are recommended.
4	Data Interoperability Requirements	To enforce data (earthquake catalogues, seismic sources, exposure, hazard and risk results, etc.) interoperability, an XML format file is to be used. XML formatted content will be exchanged over a web services-based transport mechanism. Earthquake-related parameter data is handled in QuakeML 1.1 format. For other data, we suggest that GEM develops its own, community based exchange formats (ShaML, see also the GEM1 Technical Report GEM1 Hazard: Description of input models, calculation engine and main results). Please not however that the current ShaML format is only a draft. It does not yet cover all topics required.
5	System Interoperability	*OpenGEM* interacts with other existing systems used by Hazard/Risk specialists by means of sending XML data and functionality requests

		over web services.
6	Data repository	Given the data access over web services, each system component is unaware of, and insensitive to the data persistence layer of another component, However for the data quantities expected to be handled in the *OpenGEM* framework, a high performance, high capacity, stable Database Management System with an extension for spatial data seems unavoidable. For the core *OpenGEM* repositories, we use postgreSQL with postGIS.
7	Security and Access Level	At the first level, a login ID/password will be compulsory. A multiple access level schema is required for data as well as for functionality. Access rights to functionality are handled on the portal level, while access rights for data are handled on level of the data services. *Within the OpenGEM portion developed for GEM1, only the functionality access level schema is implemented. Schemata for handling data ownership and access rights with a owner/group/public granularity are developed in the Qlarm project, which is technically closely related, but will not be further integrated into OpenGEM.*

2.3 Goals and Guidelines

The initial deployment of *OpenGEM* is within the GEM1 project. This deployment will be centrally administered. Regional Programmes like SHARE and EMME will be "linked/connected" to this central hub for shared components. They may develop and deploy local components, e.g. just relying on GEM data services, according to their specific needs, however, they are encouraged to follow the architectural and tool guidelines of *OpenGEM*, in order to have maximal code reuse and interaction capabilities.

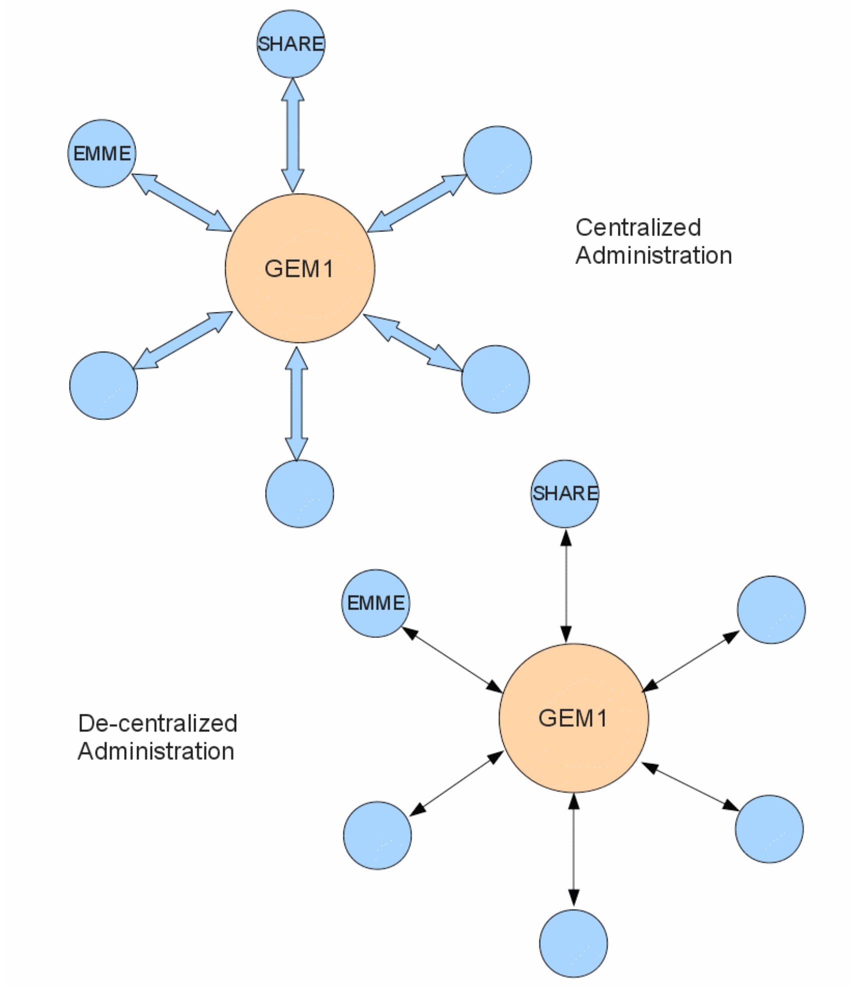


Figure 2.1 Schematic showing Centralized and De-centralized deployment administration. The link between the core "GEM Central" and satellite "Regional centers" can made strong or weak depending on the business/licensing arrangement chosen.

2.4 Development Methods

Different variants (i.e., Rapid Application Development) of Agile Methodology is to be adopted for development of different system tiers. The involved stakeholders and interaction schemes vary between system components.

1. **Database Tier**: Close interaction with the science team(s) to develop Data Models. A basic data model is adapted iteratively to represent the user requirements (rapid prototyping).
2. **Data Access Tier**: Similar to Database Tier development, include additional interactions with the core IT team (support for developing interfaces to data-dependent applications). The access tier can be developed stepwise, beginning at subsets of the data exchange standards.
3. **Application Tier**: Each application / each use case needs an interaction platform between a peer group of its users and the IT team.
4. **Portal Tier**: Stronger interactions with the end-users, but, interacts with the rest of IT team.
5. **Exchange formats**: Format development needs to be done in strong interaction with, and preferably managed by with a technically affine subset of the scientific user community. As changes in the data formats needs changes in the database tier, the application tier, as well as in the codes of most customers, Rapid prototyping is not applicable, but development cycles should be minimized in numbers, well planned, and carefully finalized.

The core part of the *OpenGEM* system is implemented as a JEE enterprise application (using a pure-JEE solution) and a portal framework built with Portlets (using Spring framework MVC). This combination is considered best to guarantee for openness, flexibility, scalability, availability of stable base packages to build on, clean oo-development, and availability of community support and specific developer knowledge on the market.

2.5 Development and Tools Strategies

- Java based coding requires Object Oriented Analysis and Development. This is supported by homogeneous use of Integrated Development Environment (Eclipse 3.4.x onwards), and tools for code modelling (UML2tools from Eclipse project), Versioning (SVN 1.6.x), Issue Tracking (trac 0.11).
- Only FOSS products are envisaged to be used. This will, of course, increase the burden of maintenance on the short term. From a long-term perspective, the Return On Investment in terms of flexibility, system expertise and freedom in usage is significant enough to justify this choice. However, only mature products with a large user base and community are advised to be chosen, to make choices future proof. Choosing products which follow industry standards will make the replacement of a product simpler, in case the product is no longer being actively enhanced or developed.
- User Interfaces are be modular and web based (with some exceptions). Modularity is being achieved by the use of Portal Frameworks (aka Portlets). Portal Frameworks are based on a Java Standards (JSR 168/286). Every major use case is be achieved using a Portlet.
- Each Application contains a suite of closely related functionality and is implemented as a separate “web application” in the application server level (JBoss 5.x).
- Deployment of the system is done in three phases: Development, Testing and Production instances. Production Instance will contain the latest GA version. Testing Instance will contain the latest RC version. Development Instance will contain the latest build. *At the end of GEM1, the system only operates development and testing, partly on non-public accessible workstations, as computing resources are still sparse and heavily used by local hazard model calculations.*
- A rudimentary user-level security will be achieved using a login/password. However, intra and inter-system security, i.e., between system components and with other system will be implemented using the Web Services

Security standards (an OASIS standard). The WS-security will be an overhead on the system performance (encrypted communication).

- A mechanism has to be set-up for the distribution of *OpenGEM and its components* to the sub-projects, in the centralized administration mode. No full-featured blueprint for this has been developed.
- The *OpenGEM* system is aimed at different communities – those with an interest in hazard, in risk, in socio-economic phenomena, or those with a combined interest. These communities have different requirements, resulting in separate database schemes and load management requirements. However, data structure and exchange formats need to be similar enough to allow easy interfacing. In all cases, To facilitate manageability of the data subsystem, besides data ownership also data origin and history is tracked, even if specific use cases do not require this.

3 System Architecture

The system caters to three major seismological communities: Hazard, Risk and Validation/Testing. Hence, the system is decomposed into three subsystems, named after the three communities. These subsystems have similar software architectures and are distributed architectures.

This system will leverage the SOA to be the architectural style, which implies the following:

- Loosely Coupled
- Standards compliant, whenever possible
- Vendor independent
- Explicit Boundaries
- Use Wire Format (not APIs), whenever possible

3.1 Hardware Architecture

Due to the heterogeneous profile of the different system components, hardware requirements are quite diverse. Aspects to take care about are:

- High availability (24x7), stability and Performance
- Extensibility
- Rapid disk access for random read/write (for databases)
- support of distributed high performance computing in a cluster setup
- support of codes with high requirements on shared memory
- Hardware support for virtualization
- Amenable to Backup and Restore functionality

These requirements are best fulfilled by a cluster of relatively large nodes with multisocket and large shared memory. The GEM modelling facility currently uses 4 SunFire X4600 M servers with 8 sockets/32 cores (upgradable to 48 cores) each and 64 GB RAM each (upgradable to 512 GB).

This hardware allows the following scaling scaling path, to be followed by each software component immediately:

1. Deployment in a virtual machine under Vmware (scalable 1..8 GB RAM and 1..4 cores). This is used for code development & testing, and a long term solution as for smaller dedicated tasks (e.g. authentication, etc.)
2. Deployment on a physical machine → used for dedicated tasks with high disk i/o and computing power requirements, e.g. the backend database for the gem1 phase, when it was more often interfaced by custom codes than by the standard opengem database abstraction layer. For the next project phase with hazard calculations operating directly on the abstraction layer, a DB cluster setup is envisaged.
3. Deployment on a loadbalanced cluster of virtual machines (used: vmware server 2.1). Envisaged for the portal, application, and webGIS servers, as well as for the testing phase of cluster-enabled calculators (see below)
4. Deployment on a cluster of physical machines Envisaged for logic tree and resampling/ monte carlo-based calculators. A cluster setup, however not done yet on the hardware of the modelling facility, has been tested with openSHA on USGS infrastructure.

Firewalling (router-based ip-individual firewalling on the ETH upstream routers) and Backup (site backup using a redundant tape roboter farm on two locations) are implemented based on infrastructure of the ETH IT services.

Project-related file server (samba), SVN/Trac code versioning/tracking, and network infrastructure are provided by SED common resources.

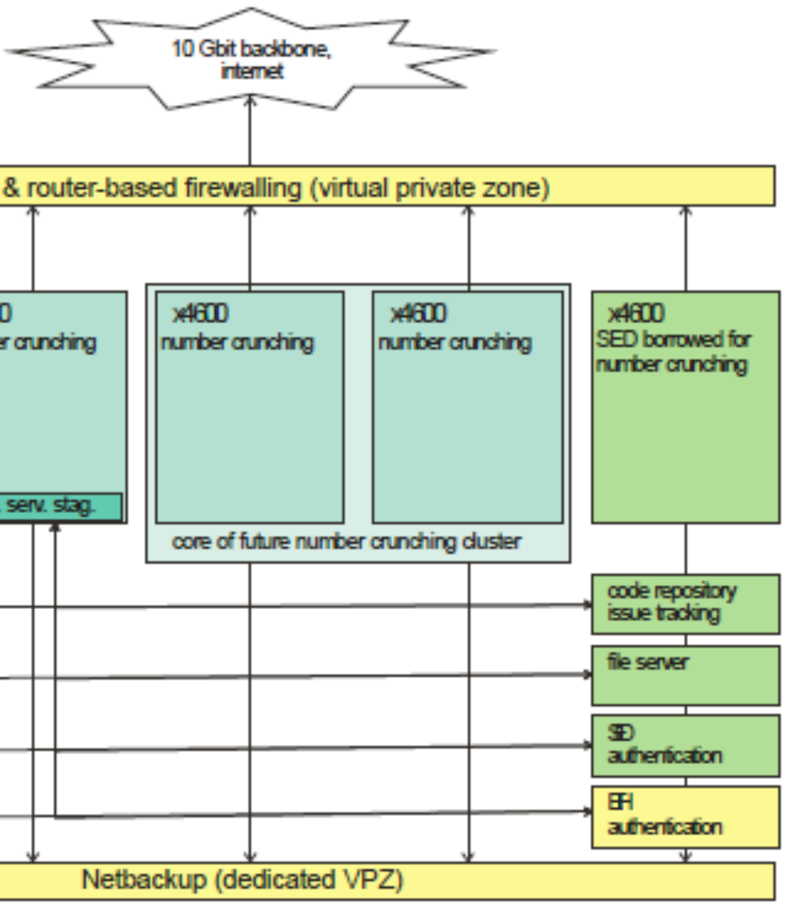


Figure 3.1 OpenGEM hardware diagram

3.2 Software Architecture Diagram

OpenGEM is an Enterprise-level application running on the PRODUCTION server, as shown in Figure 3.1. The application has been divided into multiple tiers, to better understand and separate/compartmentalize the functionality.

Database Tier: All the three sub-system have large amounts of data, which need to stored, modified, backed-up, retrieved and archived. These requirements are best achieved using a Database Management System. Additionally, since some of the data are of special category/type (i.e., Spatial), the choice of available DBMS is reduced to 3-4 only. The product "postgresql 8.3 (+ postgis extensions)" fits the requirements well and hence, has been chosen.

Data Access Tier: The data used in Earthquake modelling is available in many formats. In order to be able to use the data in the *opengem* system the data needs to be converted to a common format. Such a new format (XML based, named as PSHAML) is being proposed and created. All available data (in whichever format it is available in) needs to be convert to

PSHAML. A schematic diagram showing this flow of data is shown in Figure 3.2. In this tier, the data flow into and out-of the database will be channelized. Also, in this tier, the XML binding to Java Objects and subsequent ORM will take place.

Application Tier: The functional requirement are described as Usecases in Section 5. The application tier contains the major functionalities (in other words, the Usecases) of the system, each Usecase is mapped to a JEE Web Application.

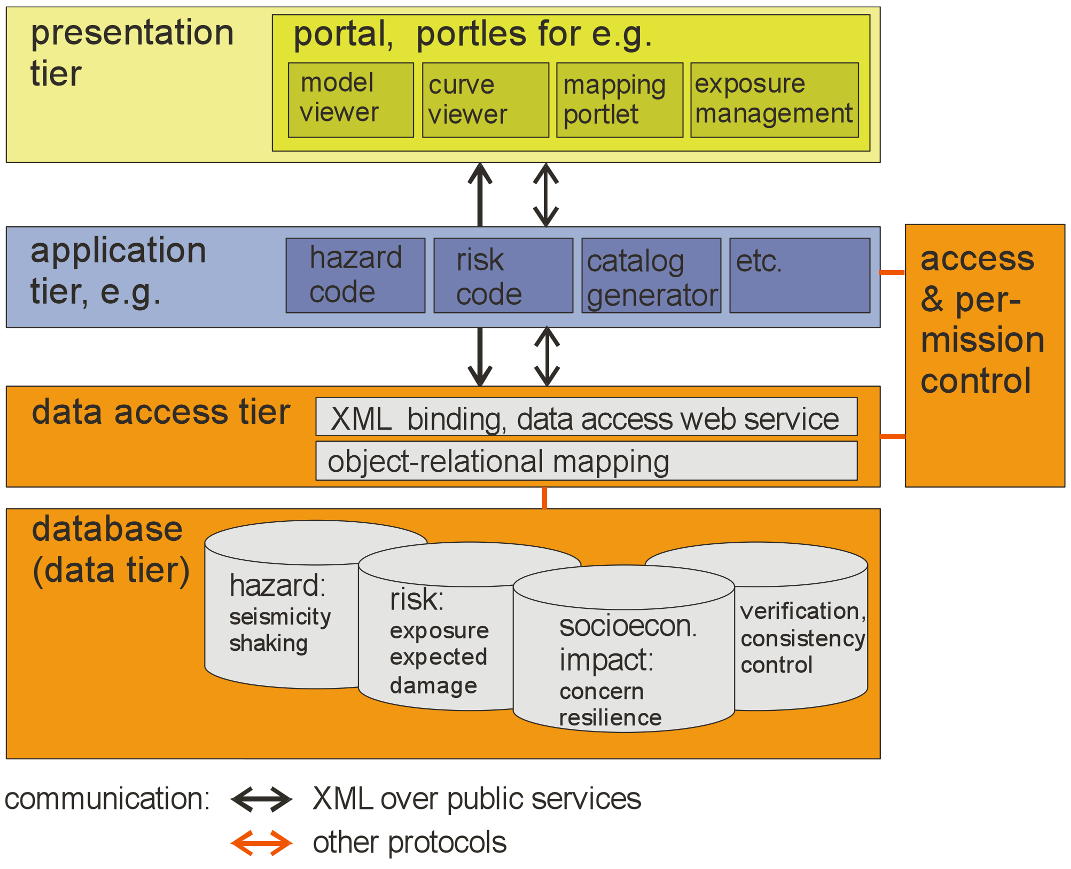


Figure 3.2 Diagram showing the overall system architecture of the OpenGEM IT system

These Web Applications exchange data (in PSHAML format using Web Services) with the entry points (for XML binding) in the Data Access Tier. All the WebApplication is grouped under the “sub-system context” or “various Engines”, which implies a clear and meaningful URLs. The Web Applications will be built using available JEE technologies and frameworks.

Presentation Tier: This tier, usually known as Web tier, is based on Portal framework (JSR 168 standards). The WebGUI is built using Portlets technology. Again, each Portlet is mapped to a Web-Application in Application tier, but, this relationship is not always 1:1.

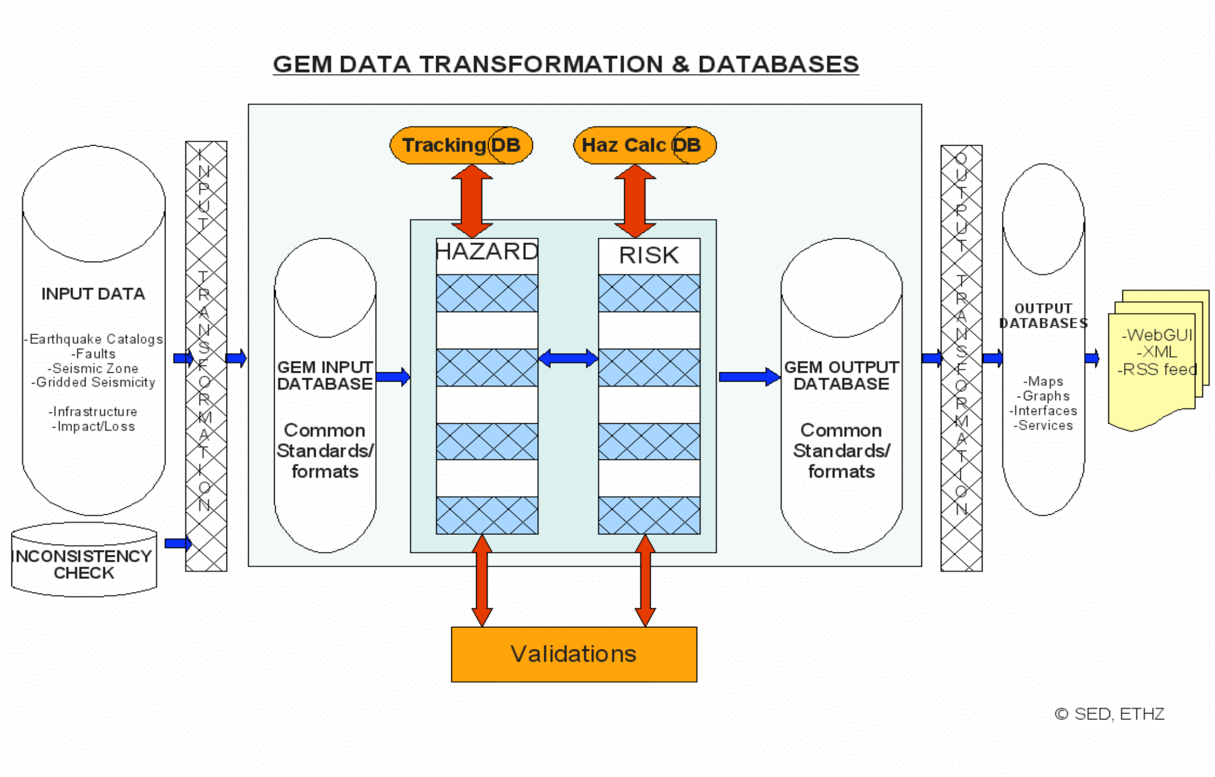


Figure 3.3 Data flow and transformation in OpenGEM system

4 Policies

4.1 Access Control

Standard access level control for services based on principally public data, standard username/password authentication and session control is used. However, if data modification services or private-owned, non public-domain datasets need to be implemented, web service security based on the OASIS 1.1 standard will be added.

In general, the following access levels are foreseen by the platform. These functionality-related access levels are managed on the portal level, by user-based allowing or disallowing portlet access.

Table 4.1 Definition of access levels, user types and available functionalities.

Level – 4	Receives static information Access to Website	Public/ Unregistered User
Level – 3	Accesses specific static databases (e.g. Single LT branch models, alternative models	Registered User/ Community Member
Level - 2	Can launch resource-demanding backend calculations submit, edit, and delete self-owned datasets	Registered Expert User
Level - 1	Has Administrator rights Configuration and write permission beyond institutional it policies	System Administrator/ IT Team Member

In addition, access levels overlap with data ownership and permissions; e.g. a level 2 user can access specific data only if he/she has reading permissions for that dataset, and a level 2 user can modify only datasets of which he/she is owner, or has been granted write access. Date-related permission control is implemented in the data access layer, i. e. server-side of the data provision services.

Within the GEM1 pilot project, access layers were only implemented on functionality levels; data ownership is prepared on database-side, but currently not checked by the data access layer. Detailed policies will need to be defined in a next stage.

4.2 Namespaces

“In general, a **namespace** is an abstract container providing context for the items (names, or technical terms, or words) it holds and allowing disambiguation of items having the same name (residing in different namespaces).” –Wikipedia

OpenGEM is a complex IT system, with many items/objects (e.g., XML format, Java packages) which requires a definition of namespaces in order to allow contributors streamlined access to prebuilt functionality and orderly introduction of their contributions. The following namespaces are promoted for *OpenGEM*. The list is subject to extension, as needs occur.

Table 4.2 Namespaces and their intended purpose.

Namespace	**Remarks**
org.opengem.xml.	XML schemata and related
org.opengem.portal.	Portal configuration, pages, layout, portlets
org.opengem.db.	database, sql, hibernate abstraction layer
Access	Jaxp classes (xml binding), hibernate-jaxp-bridge
Ws	data accessibility web services
org.opengem.codes.hazard.	Hazard codes
org.opengem.codes.risk.	Risk codes
org.opengem.codes.valid.	Validation codes
org.opengem.utils	General utilities supporting a main purpose, e.g. coordinate / unit converter
org.opengem.tools.	Tools with an independent specific purpose (e.g. logic tree offline viewer, synthetic catalogue generator etc.
org.opengem.services.	All services offered by GEM project.

4.3 System Security

- OpenGEM needs at least the following firewall exceptions:
- all systems: TCP port80: HTTP
- all systems: TCP port 443: HTTPS (authentication and OASIS-secured data transfer)
- intranet, between DB and data access layer: TCP 5432, Postgresql/postgis
- intranet, between authenticating components (portal server, data access layer) and authentication server: TCP 636, ldaps

4.4 Version Control

Versions of the following entities are controlled:

- Logical Data Model
- Software and tools developed in the GEM project
- stakeholder contributed documents and base datasets (e.g. provided by global initiatives) in original data formats
- Version control is done using subversion (http://subversion.apache.org/), and trac (http://trac.edgewall.org/) for issue tracking in the software development.

4.5 Capacity Planning Requirements

From the overall design, the OpenGEM hardware and software is fairly scalable. However, a full capacity planning for hazard and risk calculations, as well as for the end-user triggered load of the web exposed components, are pending tasks (which also depend on the future scientific work plan).

4.6 Licensing

As *OpenGEM* is expected to be a FOSS product, thus an open source licensing model, as e.g. “GNU General Public License, version 3”, may, because of its openness and compatibility with other open source licensing systems, be most appropriate for OpenGEM-developed codes. However, as the licensing also depend on the licenses of 3rd party included libraries, and those are not always defined yet in every case.

The licensing of GEM results in terms of data is a separate issue. However, with moving into owned data, decisions may be taken on a dataset level (also considering the licensing coming along with the input datasets).

5 User-System Interactions

The following use cases have been identified:

1. Worldwide data users of different communities:

a) look up hazard curves, and hazard spectra

b) look up hazard maps

c) look up risk maps – probabilistic and deterministic

d) look up risk curves

e) disaggregate hazard results

f) explore hazard and risk models, and their input data (especially i. seismic sources, and ii. Exposure data) – with respective tabulation and mapping

g) user support materials and interaction

All these services need proper portlet-mounted frontends. Case studies have been prepared within GEM1 for 1a (hazard curves), 1b, and 1c. However, the use cases are not fully specified. The best specified one, at least from the frontend point of view, are 1a and 1b. Detailed frontend specifications are included in Appendix B. These specifications were developed together, and mostly driven by the needs, of the SHARE Regional Programme. They are assumed to be technically in sync with the need for the OpenGEM portal, however, the look and feel of the figures of the appendix follows the SHARE requirements. For technical specifications of the portlet-mounted applications, as far as available by now, see section 6.5.

2. Scientists working in the field of hazard & risk modelling:

h) explore logic tree structures

i) compile/edit logic tree structures

j) submit seismic source information

k) submit exposure datasets /

l) configure and trigger hazard, and risk calculations based on public, and own input datasets

m) subscribe proper codes to data services (frontend client-side)

n) publish/subscribe rss, communicate between OpenGEM users, integrate in social networks and related communication systems.

For the preliminary results developed in GEM1, hazard and risk calculations have been created using locally accessed installations of non-portal-integrated calculators; the development of detailed specifications, and the implementation of the portal-mounted applications will need to happen in the framework of the GEM-MF. Use cases for testing and verification are not defined yet.

3. GEM collaborators

o) workflow for reviewing, approving and integrating base data into the authorized & public GEM base datasets

p) workflow

4. Administrative staff

q) user & rights system & code deployment administration

r) user & rights administration

s) system use, load, and state of health monitoring

These use cases are mostly needed inside the GEM-MF. However, as the staff is distributed over multiple sites, frontends should also be web-mounted. Detailed specifications of these applications and their interface should not impact the OpenGEM user, they are only of GEM-MF internal relevance.

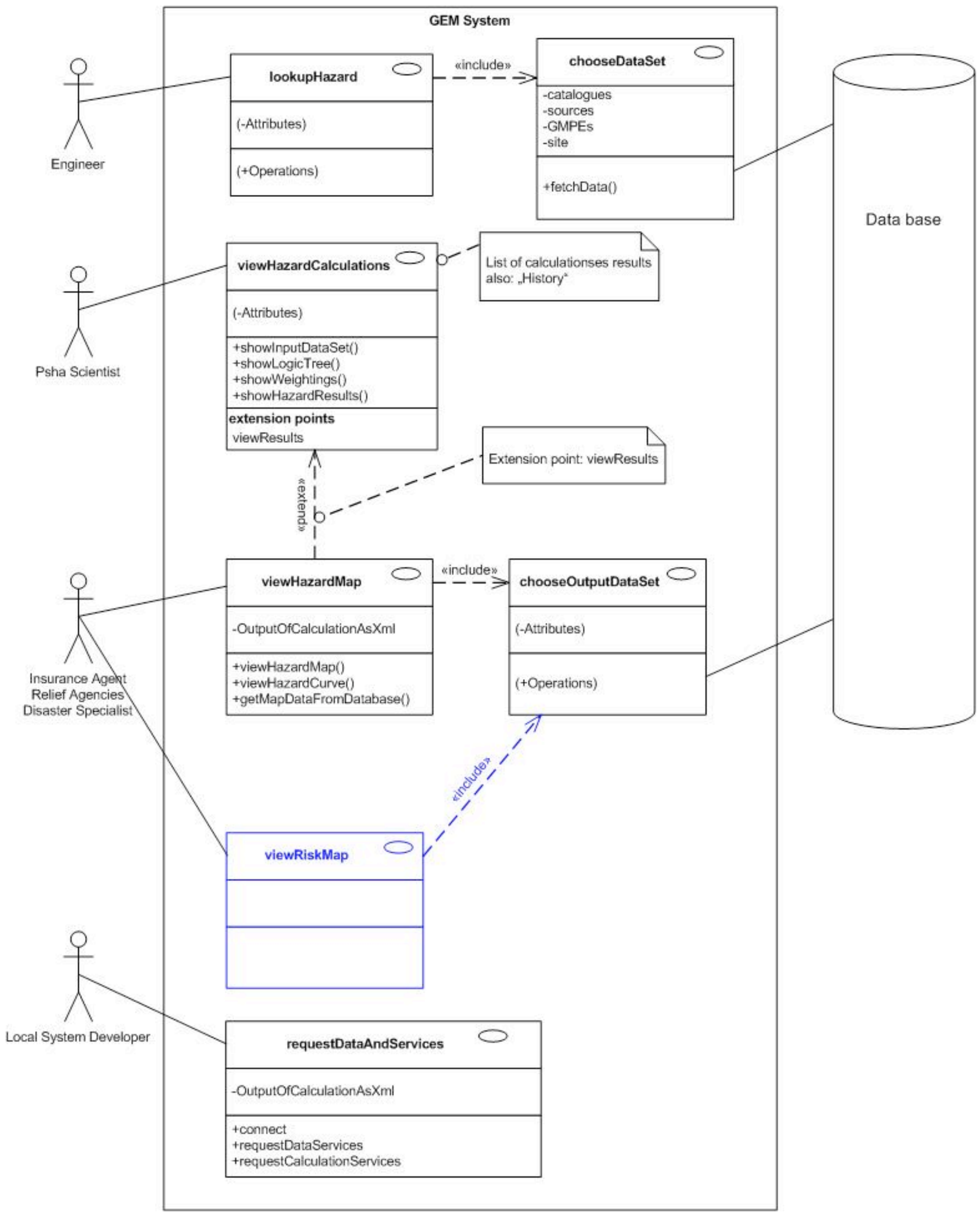


Figure 5.1 Usecase diagram, Part-I

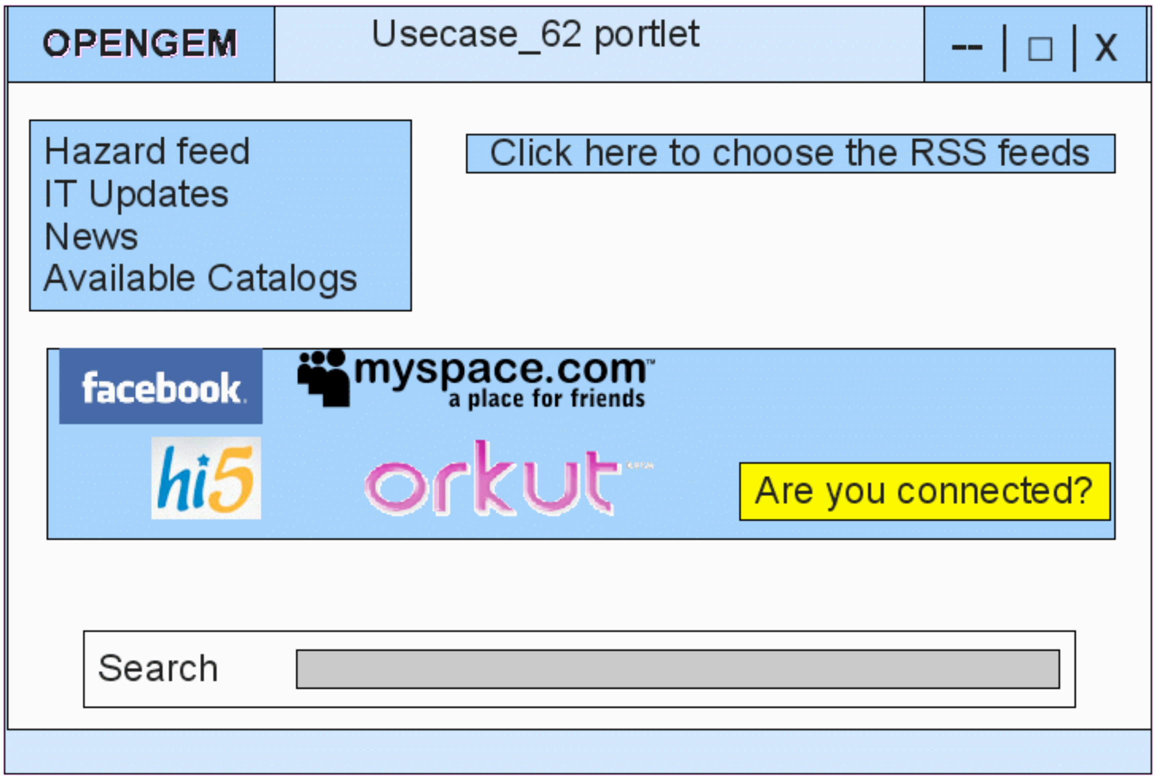


Figure 5.2 Possible community collaboration interface

6 Detailed Technical System Design of the OpenGEM Development System (GEM1 level)

A detailed system design of the *OpenGEM* system, corresponding to the Use cases in Section 5, is presented here. The major components are

- Database server
- JEE Application server, containing the Servlet container
- Enterprise Information Portal, containing the Portlet container
- LDAP server
- Map server
- Web Services - * (Interoperability, Security)

The Database server (**gemdsrvr.ethz.ch:5432 for development, gemsun01.ethz.ch for testing**) is running Postgresql 8.3 (with postgis extensions). The postgis extensions are essential for handling spatial information. The database servers are backed up on the ETH Zürich roboter, every evening.

JBoss 5.1.x, an JEE application server (VM **gemasrvr.ethz.ch:8080 for testing, local machines for development**) Per use case, one web application with portlet frontend is deployed).

Jetspeed 2.2.x, is an Enterprise Information Portal (**gempsrvr.ethz.ch:8080**) has an in-built Portlet container (Pluto 2.0, JSR-286 complaint). Each of these portlets contain the necessary GUI for data/user input and output.

An LDAP server (openLDAP 2.4.x) (**gemldap.ethz.ch:389,686**) will provide the required Access Control mechanism and forms a common user/resource directory.

UMN MAPSERVER 5.4 is a widely used open source mapserver product. The *opengem* mapserver (**gemmsrvr.ethz.ch**) will provide the maps on-demand, directly based on primary data in the database. In client applications, the geoext / openlayers javascript libraries are the basis to provide mapping functionality to the users.

Web Service protocols are to be used to achieve SOA. It is used to exchange data between: Portlets (Presentation tier) ↔ Web-Application(Application tier), Web-Application (Application tier) ↔ entry points for XML-binding (Data Access tier) and for Data Services. The open source product Axis2 (Apache Software Foundation) “Axis2 1.5” is to be used for WS-I and WS-Security protocols.

6.1 Functional Structure

6.1.1 Hazard Sub-System

The Hazard Subsystem is one of the major subsystems of the *OpenGEM* system, the other two being the Risk Subsystem and the Validation Subsystem. The Hazard Subsystem handles functions related to the conversion and maintenance of data, the definition and maintenance of hazard input models, the processing and maintenance of hazard calculations, the generation of hazard outputs as well as the maintenance of supporting information. Figure 6.2 shows the overall functional structure of the Hazard Subsystem which is supported by the *OpenGEM* IT Project Infrastructure.

There are five functional modules in the Hazard Subsystem, as follows:

- Data Conversion Component
- Data Maintenance Component
- Hazard Input Model Definition Component
- Hazard Calculation Processing Component
- Hazard Output Component

The **Data Conversion Component** handles the conversion of outside files into the GEM database structure formats. This includes the following modules:

- Earthquake Catalogue and Events Conversion Module
- Source Geometry Catalogue and Seismic Sources Conversion Module

The Earthquake Catalogue and Events Conversion Module converts earthquake catalogues and the events stored in given catalogs into the GEM database structure. The Source Geometry Catalogue and Seismic Sources Conversion Module involves the conversion of Source Geometry Catalogs and the seismic source data in the specific catalogs. These seismic source data may include area sources, faults, gridded seismicity points and subduction faults and their values for specific magnitude frequency distributions.

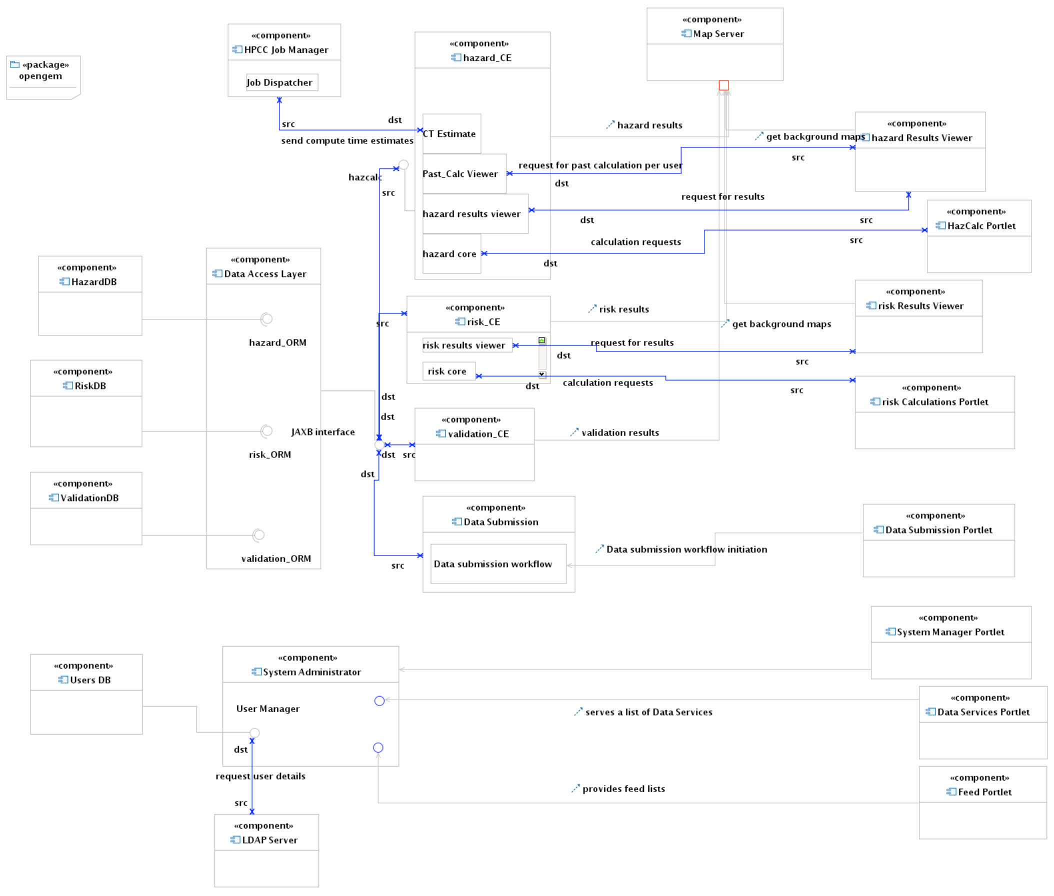


Figure 6.1 A detailed component of the OpenGEM system

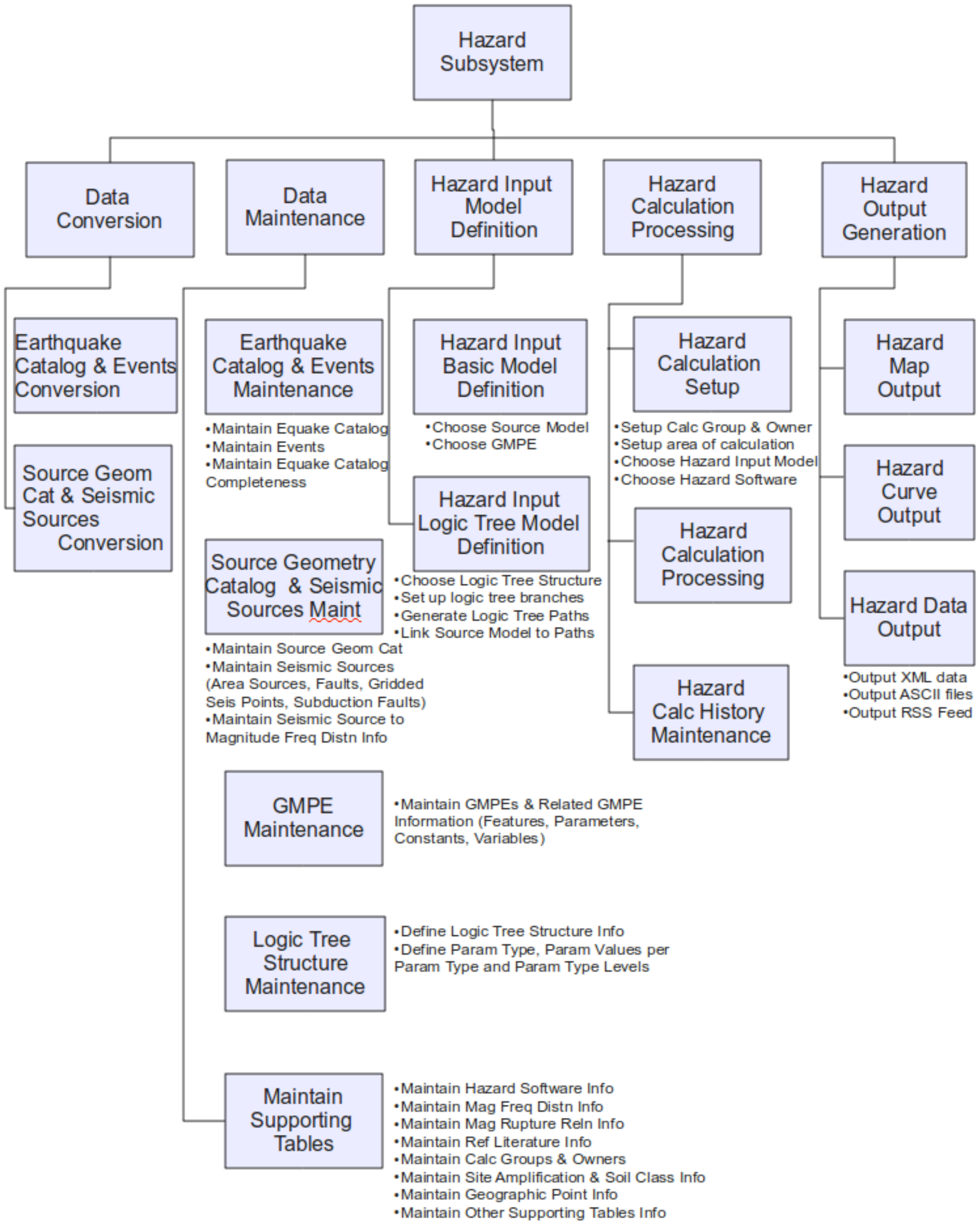

Hazard Subsystem
Data Conversion
Data Maintenance
Hazard Input Model Definition
Hazard Calculation Processing
Hazard Output Generation
Earthquake Catalog & Events Conversion
Source Geom Cat & Seismic Sources Conversion
Earthquake Catalog & Events Maintenance
•Maintain Equake Catalog
•Maintain Events
•Maintain Equake Catalog Completeness
Source Geometry Catalog & Seismic Sources Maint
•Maintain Source Geom Cat
•Maintain Seismic Sources (Area Sources, Faults, Gridded Seis Points, Subduction Faults)
•Maintain Seismic Source to Magnitude Freq Distn Info
GMPE Maintenance
•Maintain GMPEs & Related GMPE Information (Features, Parameters, Constants, Variables)
Logic Tree Structure Maintenance
•Define Logic Tree Structure Info
•Define Param Type, Param Values per Param Type and Param Type Levels
Maintain Supporting Tables
•Maintain Hazard Software Info
•Maintain Mag Freq Distn Info
•Maintain Mag Rupture Reln Info
•Maintain Ref Literature Info
•Maintain Calc Groups & Owners
•Maintain Site Amplification & Soil Class Info
•Maintain Geographic Point Info
•Maintain Other Supporting Tables Info
Hazard Input Basic Model Definition
•Choose Source Model
•Choose GMPE
Hazard Input Logic Tree Model Definition
•Choose Logic Tree Structure
•Set up logic tree branches
•Generate Logic Tree Paths
•Link Source Model to Paths
Hazard Calculation Setup
•Setup Calc Group & Owner
•Setup area of calculation
•Choose Hazard Input Model
•Choose Hazard Software
Hazard Calculation Processing
Hazard Calc History Maintenance
Hazard Map Output
Hazard Curve Output
Hazard Data Output
•Output XML data
•Output ASCII files
•Output RSS Feed

Figure 6.2 Hazard subsystem, functional structure diagram

The **Data Maintenance Component** handles the maintenance of GEM-related data including the addition, modification and deletion of records. This component includes the following modules:

- Earthquake Catalogue and Event Maintenance Module
- Source Geometry Catalogue and Seismic Source Maintenance Module
- GMPE Maintenance Module

- Logic Tree Structure Maintenance Module
- Supporting Tables Maintenance Module
- The Earthquake Catalogue Maintenance Module handles the maintenance of earthquake catalogues including their corresponding events and a description of their completeness in terms of area and/or time.

The Source Geometry Catalogue Maintenance Module handles the maintenance of source geometry catalogues and their corresponding Seismic Sources, particularly, area sources, faults and their fault characteristics, gridded seismicity points, and/or subduction faults. Each seismic source may have more than one magnitude frequency distribution. Corresponding values related to a seismic source for a particular magnitude frequency distribution are stored in the database, i.e. a and b values for seismic sources with Gutenberg-Richter distribution; characteristic magnitude and characteristic rate for seismic sources with Characteristic distribution. Discrete distributions are also supported, with the database storing data on bin width, number of bins and magnitude/seismicity rate pairs.

The GMPE Maintenance Module maintains Ground Motion Prediction Equations (also known as Attenuation Relations) and related information.

The Logic Tree Structure Maintenance Module handles the description of available logic tree structures including the set-up of Parameter Types (for example, Fault Model, Dip-Uncertainty Models) and the corresponding Parameter Values that are possible for each parameter type (for example, Characteristic and Gütenberg-Richter for the Fault Model Parameter Type). In addition, the logic tree level for each parameter type is also defined for a given logic tree structure.

The Supporting Tables Maintenance Module handles the maintenance of other data supporting GEM processes, such as Hazard Software Information, Magnitude Frequency Distribution Information, Magnitude Rupture Relation Information, Seismotectonic Environment Information, Intensity Measure Type Information, Site Amplification Information, Soil Class Information, Reference Literature Information, Calculation Group and Calculation Owner Information, Geographic Point Information and other supporting tables.

The **Hazard Input Model Definition Component** sets up and tracks Hazard Input Models which could either be a Basic Model Definition or a Logic Tree Model Definition. This will include a Hazard Input Basic Model Definition Module and a Hazard Input Logic Tree Model Definition Module.

The Hazard Input Basic Model Definition involves the definition of a simple model for calculating hazard without the use of a logic tree. This includes the choice of a source model from the set of source geometry catalogues available and the choice of a GMPE. The Hazard Input Logic Tree Model Definition allows for the definition of more complex models using logic trees. Necessary information for this model includes the choice of a logic tree structure, the set-up of the characteristics of each logic tree branch (i.e. the specification of Parameter type-Parameter Value-Branch Weight triple), the generation of the corresponding logic tree paths and the linking of a source model to each logic tree path in the given hazard input logic tree model.

The **Hazard Calculation Processing Component** handles actual hazard calculation processing and includes the following modules:

- Hazard Calculation Setup Module
- Hazard Calculation Processing Module
- Hazard Calculation History Maintenance Module

The Hazard Calculation Setup Module involves specifying the area of the calculation, choosing the Hazard Input Model to use, and specifying the hazard software to be used for the calculation. The Hazard Calculation Processing Module involves the actual processing of the hazard calculation. The Hazard Calculation History Maintenance Module involves the actual tracking of past calculations.

The **Hazard Output Component** handles the data output requirements of the Hazard Component of the GEM system. This includes the following modules:

Hazard Map Output Module

Hazard Curve Output Module

Hazard Data Output Module

The Hazard Map Output module involves the generation of the hazard maps. The Hazard Curve Module handles the generation of hazard curves. The Hazard Data Output module handles the output of Hazard Data in other forms such as XML data, ASCII files, RSS feeds and so on.

6.2 Database Design

The database of the GEM System is designed to handle the complexity involved in hazard calculation and processing requirements. The use of relational modelling techniques to come up with the final model ensures data independence, consistency and reliability of data.

OpenGEM requires data models and database sections for Earthquake Hazard Input / Modelling / Output data, in the Hazard section, Exposure, modelling and Risk/Loss output data for the Risk section. Additional requirements may arise from unit testing and verification tools.

However, within GEM1, only for hazard a full-featured database scheme for end-to-end use was developed. Given the late start of the Risk calculator job definition and software development, we decided operating risk calculations on simplified file-based data storage. All further discussions of database schemes refer only to hazard data; early drafts of the risk database are not presented, as they base on incomplete specification of the risk calculations and will still undergo serious structural changes.

The Entity Relationship Diagram (ERD) describing the GEM Hazard Data Model is shown in Figure 6.3. This ERD is realized in the physical table layouts that are discussed in the Appendix.

The GEM database is implemented in Postgresql 8.3, a powerful open-source community-supported Relational Database Management System (RDBMS). To handle geographic/spatial data, Postgis extensions are introduced into the system. Hibernate is used as an Object-Relational (OR) Mapping tool to be able to achieve an independent code base and at the same time maintain the Object Oriented paradigm in the face of the relational paradigm present in the GEM relational database structure. Object-relational mapping is used by the data-access layer, however it is cached from the application layer. The application layer is served by the data access layer with data objects in XML, furnished over web services (establishing an xml format family intended to be developed into a community data exchange standard within the further development of the GEM project). Thus, applications do not need to be aware of the relational backend of opengem, nor of its technical implementation.

Major subsets of stored data, covering major functionalities required by the opengem System are discussed in more detail in the succeeding sections.

6.2.1 Hazard Data: Data Objects and Resultant Data Structures

- The dataset, covering functionalities required by the *opengem* system, are as follows:
- Earthquake Catalogues and Events
- Source Geometry Catalogues and Seismic Sources
- GMPE Definition
- Logic Tree Structure Definition
- Hazard Input Basic and Logic Tree Model Definition
- Hazard Calculation Process
- Hazard Output
- Reference Literature

The physical table layouts of the *OpenGEM* system are presented in Appendix-D and the table layouts are ordered alphabetically by table name.

Earthquake Catalogues and Events

<u>Tables</u>

EARTHQUAKECATALOG - Earthquake Catalogue Table

EQCATCOMPLETENESS – Earthquake Catalogue Completeness Table

EVENT – Event Table

<u>Description</u>

This set of tables keep track of earthquake catalogue and event information. The Earthquake Catalogue Table (EARTHQUAKECATALOG) stores information regarding earthquake catalogues that are in the system. Earthquake catalogues contain events as stored in the Event table (EVENT). These events have information such as location, time of occurrence, magnitude size. The Earthquake Catalogue Completeness Table (EQCATCOMPLETENESS) describes the completeness of an earthquake catalogue in terms of either time or space/area.

Source Geometry Catalogues and Seismic Sources

<u>Tables</u>

SOURCEGEOMETRYCATALOG – Source Geometry Catalogue Table

SEISMICSOURCE – Seismic Source Table

SSOURCEMFD – Seismic Source - Magnitude Frequency Distribution Table

SFAULTCHAR – Seismic Fault Characteristics Table

SEISMOTECENVT – Seismotectonic Environment Table

MAGFREQDISTN - Magnitude Frequency Distribution Table

MAGRUPTURERELATION – Magnitude Rupture Relation Table

<u>Description</u>

This set of tables keep track of Source Geometry Catalogues, Seismic Sources and related seismic source information. The Source Geometry Catalogue Table (SOURCEGEOMETRYCATALOG) keeps track of information on source geometry catalogues that are in the system. A source geometry catalogue may have one or more seismic sources associated with it as stored in the Seismic Source Table (SEISMICSOURCE). These seismic sources may either be area sources, faults, gridded seismicity points and/or subduction faults. They may also have possible geometric/spatial data types. Generally, area sources are in the form of polygon or multipolygon; faults in the form of multilinestring; gridded seismicity in the form of a point; and subduction faults in the form of a top multilinestring and bottom multilinestring. Each seismic source may have one or more Magnitude Frequency Distribution types. This relationship from Seismic Source to Magnitude Frequency Distribution is specified in the Seismic Source Magnitude Frequency Distribution Table (SSOURCEMFD), such that this table contains frequency distribution values relevant for a seismic source, as well as magnitude rupture relations. For example, a fault may be described with the Gutenberg Richter Distribution, and will have values for a, b, maximum magnitude, minimum magnitude and so on, whereas a fault defined for the Characteristic Distribution will have values for characteristic magnitude and characteristic rate. Discrete distributions may also be described with corresponding data for bin widths, number of intervals and magnitude-seismicity rate pairs. The Magnitude Frequency Distribution Types, such as Gutenberg-Richter and Characteristic Distributions, are defined in the Magnitude Frequency Distribution Table (MAGFREQDISTN). The Magnitude Rupture Relation types possible for a given seismic source are stored in the Magnitude Rupture Relation Table (MAGRUPTURERELATION). These may have values such as, for example, Wells and Coppersmith 1994. The Seismic Fault Characteristics Table (SFAULTCHAR) contains additional information specific to

Faults, such as slip rate, floating rupture type and so on. The Seismic Source may also have a particular seismotectonic environment, types of which are as described in the Seismotectonic Environment Table (SEISMOTECENVT).

GMPE Definition

Tables

GMPE – Ground Motion Prediction Equation (GMPE) Table

GMPEFEATURE – GMPE Feature Table

GMPEPARAMETER – GMPE Parameter Table

EVARIABLE - Equation Variable Table

ECONSTANT – Equation Constant Table

GEFEAUTUREVALUE – GMPE Feature Value Table

GMPARAMVALUE – GMPE Parameter Value Table

GMEVARIABLE - GMPE Variable Value Table

GMECONSTANT - GMPE Constant Value Table

Description

This set of tables keep track of Ground Motion Prediction Equation (GMPE) Information, also sometimes called Attenuation Relations, in the GMPE Table (GMPE). Each GMPE may have different features. Possible features are tracked in the GMPE Feature Table (GMPEFEATURE). Examples of possible GMPE features are Minimum Magnitude, Maximum Magnitude, Regression methods and others. Feature values for a particular GMPE are defined in the GMPE Feature Value Table (GEFEATUREVALUE). Each GMPE may also have different parameters as defined in the GMPE Parameter Table (GMPEPARAMETER). These provide possible default settings for particular GMPEs as defined in the GMPE Parameter Value table (GMPARAMVALUE). Each GMPE has associated Variable with values and Constants with specified values as kept in the Ground Motion Prediction Equation Variable Table (GMEVARIABLE) and Ground Motion Prediction Equation Constant Table (GMECONSTANT). Supporting tables which describe possible Equation Variables and Constants that may be part of an equation are in the Equation Variable Table (EVARIABLE) and Equation Constant Table (ECONSTANT).

Note: at the time given, the database just tracks/documents the GMPE. However, each GMPE also needs to be implemented in the hazard calculator. An alternative would be to create a generic GMPE calculator, featured with a formula parser accessing the database. However, as the to be expected complexity of GMPEs was unknown at design time, we have not implemented this feature.

Logic Tree Structure Definition

Tables

LOGICTREESTRUC – Logic Tree Structure Table

LTREEPARAMTYPE – Logic Tree Parameter Type Table

LTREEPARAMVALUE – Logic Tree Parameter Value Table

LTREEPARAMTYPELEVEL – Logic Tree Parameter Type Level Table

Description

This set of tables describe Logic Tree Structures that are kept by the system. The Logic Tree Structure Table (LOGICTREESTRUC) describes a particular logic tree structure and its characteristics. The Logic Tree Parameter Type Table (LTREEPARAMTYPE) stores the possible Logic Tree Parameter Types that may be used in a logic tree structure definition, for example, Fault Model or Dip-Uncertainty Model. The Logic Tree Parameter Value Table

(LTREEPARAMVALUE) defines the possible Logic Tree Parameter Values for a given Logic Tree Parameter Type. For example, Characteristic Distribution or Gutenberg-Richter Distribution are possible logic tree parameter values for the Fault Model Parameter Type. The Logic Tree Parameter Type Level Table (LTREEPARAMTYPELEVEL) defines the specified parameter type for a given Logic Tree Level. The root of a logic tree is always level 0, the next level is defined as level 1 and so on until level n. For example, for the USGS NSHM 2008 WUS Model, the first level of the logic tree has the Fault Model Parameter Type, the second level contains the Dip-Uncertainty Model Parameter Type and so on.

Hazard Input Basic and Logic Tree Model Definition

<u>Tables</u>

HAZARDINPUTBASICMODEL – Hazard Input Basic Model Table

HAZARDINPUTLTREEMODEL – Hazard Input Logic Tree Model Table

HILMPATH – Hazard Input Logic Tree Model Path Table

HILMRULESET – Hazard Input Logic Tree Model Ruleset Table

HIBMGE – Hazard Input Basic Model GMPE Table

HILMPGE – Hazard Input Logic Tree Path Model GMPE Table

<u>Description</u>

This set of tables specifies the definition of the Hazard Input Calculation Model or a model that may be used for Hazard Calculation. The Hazard Input Calculation Model may either use the Hazard Input Basic Model (no logic tree defined) or the Hazard Input Logic Tree Model (uses a logic tree). The Hazard Input Basic Model Table (HAZARDINPUTBASICMODEL) may have, among other information, a specified region for processing, an associated source geometry catalogue and a chosen GMPE. The Hazard Input Logic Tree Model Table (HAZARDINPUTLTREEMODEL) specifies a set of logic tree paths, based on a logic tree structure, that is used for calculation. The Hazard Input Logic Tree Model Path Table (HILMPATH) specifies each logic tree path contained in the Hazard Input Calculation Logic Tree Model. Along each logic tree path, branch data is stored, in particular, parameter type-parameter value-branch weight information. Each logic tree path has a particular source model or source geometry catalogue with associated seismic sources that is associated with it. The Hazard Input Logic Tree Ruleset Table (HILMRULESET) specifies the rules associated with each branch in a particular logic tree path. In particular, it specifies the coverage of each branch (all sources or specified sources), the action to be done (replace value, add value), and other information relating to logic tree branches. The Hazard Input Basic Model GMPE Table (HIBMGE) describes the GMPEs that are to be used in a hazard input basic model implementation and weight values of each GMPE. The Hazard Input Logic Tree Path Model GMPE Table (HILMGE) describes the GMPEs that are to be used in a logic tree implementation and weight values of each GMPE for a specified logic tree path.

Hazard Calculation Process

<u>Tables</u>

HAZARDCALCULATION - Hazard Calculation Table

HAZARDSOFTWARE – Hazard Software Table

CALCULATIONOWNER – Calculation Owner Table

CALCULATIONGROUP – Calculation Group Table

<u>Description</u>

This set of tables describe the information necessary for Hazard Calculation Processing. The Hazard Calculation Table (HAZARDCALCULATION) stores details of calculations that were generated by the system or that come from other sources, in terms of name, start and end time of calculation, the hazard input calculation model definition used (as specified in the *Hazard Input Basic and Logic Tree Model Definition*), and other related information. This table also keeps track of the specific hazard software that was used for processing, if any, as stored in the Hazard Software Table (HAZARDSOFTWARE). In addition, the originator of the calculation is also tracked in Hazard Calculation Processing through a relation to the Calculation Owner table (CALCULATIONOWNER) which is in turn related to the Calculation Group Table (CALCULATIONGROUP). CALCULATIONGROUP can have one or more Calculation Owners.

Hazard Output

Tables

HAZARDMAP - Hazard Map Table

HAZARDCURVE – Hazard Curve Table

HAZARDPOINTVALUE – Hazard Point Value Table

GEOPOINT – Geographic Point Table

INTENSITYMEASURETYPE – Intensity Measure Type Table

SOILCLASS – Soil Class Table

SITEAMPLIFICATION – Site Amplification Table

Description

This set of tables describe the Hazard Output resulting from Hazard Calculation Processing. This is stored in the Hazard Map table (HAZARDMAP) that tracks hazard map information including a description of the generated hazard map, its probability of exceedance, return period and other information necessary. A hazard map can have one or more hazard point values as specified in the Hazard Point Value Table (HAZARDPOINTVALUE). Hazard output can also be in the form of Hazard Curves which are stored in the Hazard Curve table (HAZARDCURVE). This table tracks hazard curve information including the geographic point of interest, the maximum and minimum ground motion and other information necessary to describe a hazard curve. A hazard curve is described by one or more hazard point values as specified in the Hazard Point Value Table (HAZARDPOINTVALUE). There are several intensity measure types which may be defined for a given hazard point value. The possible intensity measure types are stored in the Intensity Measure Type Table (INTENSITYMEASURETYPE). Each hazard point value may have one or more intensity measure types. Each hazard point has a corresponding specific geographic point as stored in the Geographic Point Table (GEOPOINT). This geographic point has Site Amplification characteristics, i.e. VS30, NEHRP classification, as specified in the Site Amplification Table (SITEAMPLIFICATION). A geographic point may also have soil class characteristics, i.e. SIA 261, as stored in the Soil Class Table (SOILCLASS).

Reference Literature

Tables

REFERENCELITERATURE – Reference Literature Table

ECREFERENCE – Earthquake Catalogue Reference Literature Table

SCREFERENCE – Source Geometry Catalogue Reference Literature Table

GEREFERENCE– GMPE Reference Literature Table

HIBMREFERENCE– Hazard Input Basic Model Reference Literature Table

HILMREFERENCE– Hazard Input Logic Tree Model Reference Literature Table

Description

This set of tables tracks Reference Literature in the Reference Literature Table (REFERENCELITERATURE) as related to some components of hazard calculation processing. The components with possible reference literature information include earthquake catalogues tracked in the Earthquake Catalogue References Table (ECREREFENCE), the source geometry catalogues tracked in the Source Geometry Catalogue References Table (SCREFERENCE), the GMPEs tracked in the GMPE References Table (GEREFERENCE), the hazard input basic model tracked in the Hazard Input Basic Model References Table (HIBMREFERENCE), and the hazard input logic tree model tracked in the Hazard Input Logic tree Model References Table (HILMREFERENCE).

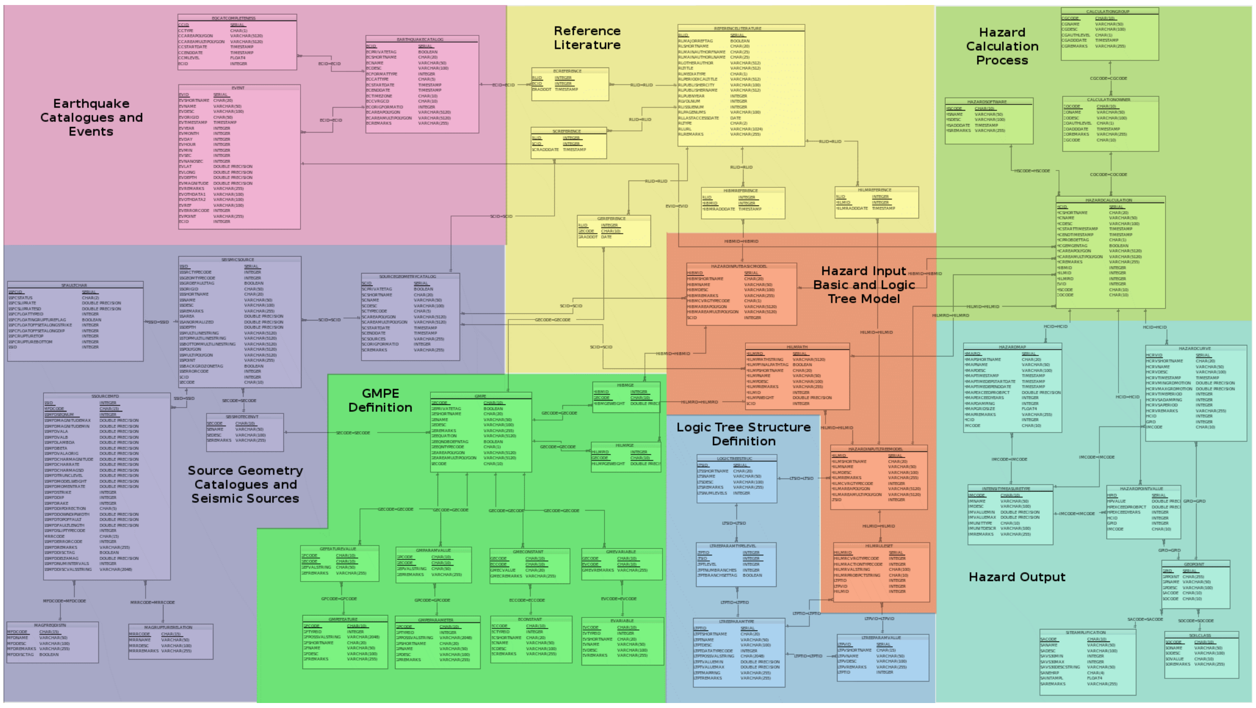

Earthquake Catalogues and Events
Reference Literature
Hazard Calculation Process
Hazard Input Basic and Logic Tree Model
Source Geometry Catalogues and Seismic Sources
GMPE Definition
Logic Tree Structure Definition
Hazard Output

Figure 6.3 GEM Database Structure

6.3 Data Access Layer

The *OpenGEM* system will use object-oriented technology such as Java to build the application software and relational database (Postgresql 8.3) to store the data, which are by far the norm. There is a need to deal with the object relational (O/R) “impedance mismatch”, and the following two exercises needs to be done: the process of mapping objects to relational databases and how to implement those mappings. In this document the term “mapping” will be used to refer to how objects and their relationships are mapped to the tables and relationships between them in a database.

Since *OpenGEM* will be persisting data in a Postgresql Database, the OR Mapping tool “Hibernate 3.3.x” along with “Hibernate Spatial 1.0” is to be used. A Schematic of the Hibernate Architecture is shown in Figure 6.4. For *OpenGEM*, JDBC type transaction demarcation is being used.

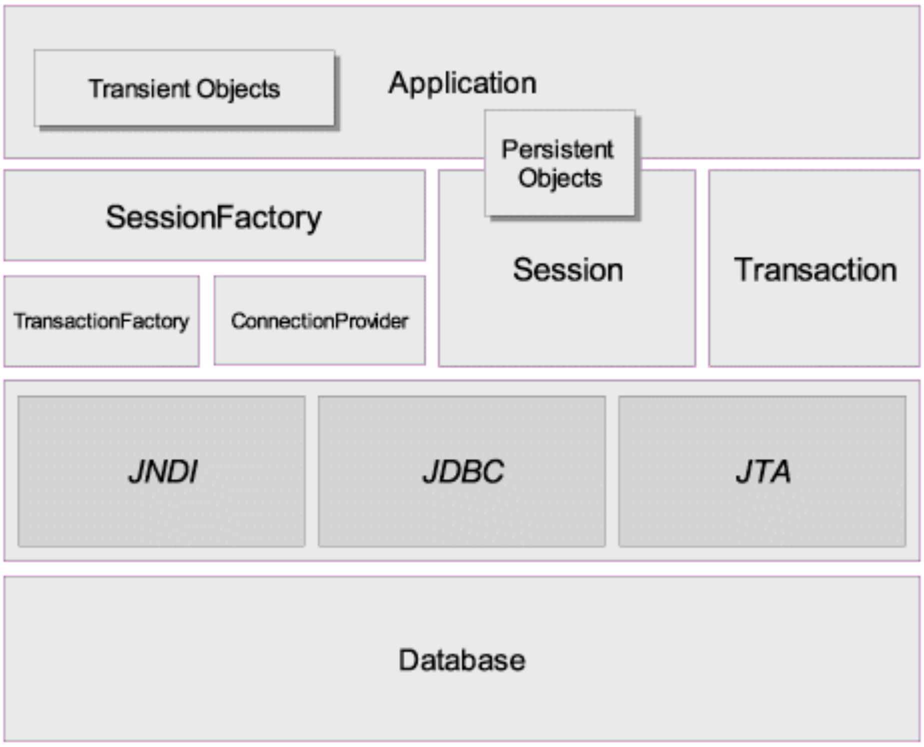


Figure 6.4 The Hibernate architecture

Adhering to the principles of SOA, web services is used to transport XML files/messages across different software components of *OpenGEM* system. This necessitates the use of "XML Data Binding" mechanisms. Since Hibernate offers limited XML mapping functionality, JAXB2 reference implementation from SUN is to be used "on-top" of the OR Mapping tool. The various entities in this layer are shown in Figure 6.5.

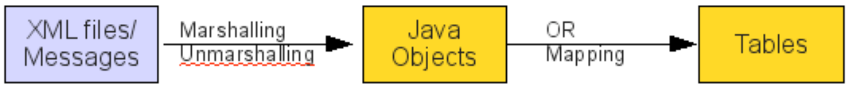


Figure 6.5 Various entities in the opengem Data Access Layer

A detailed OpenGEM Data Access Layer Architecture is shown in Figure 6.5. There are two issues that are additionally considered, apart from the basic OR Mapping considerations.

Mapping Structure: Each XML element is mapped to one table in the database. This facilitates the use of "XML Schema generators". However, mapping an attribute of an XMl element to a table in the database, would eventually reduced the file size of the XML data files, but, would imply manual rework of the automatically generated XML Schemas

Sessions/Transactions: This relates to the long user interaction/conversations with the system, as an example Hazard Calculation Portlets. Every time a user evokes/starts/interacts with the Portlet, a HTTP session is generated. The data generated in each session may have to be persisted into the database. There are two strategies for this:

Session-per-request-with-detached-objects: Once persistent objects are considered *detached* during user think-time and have to be reattached to a new Session after they have been modified.

Session-per-conversation: In this case a single Session has a bigger scope than a single database transaction and it might span several database transactions. Each request event is processed within a single database transaction, but flushing of the Session would be delayed until the end of the conversation and the last database transaction, to make the conversation atomic. The Session is held in disconnected state, with no open database connection, during user think-time. Hibernate's automatic optimistic concurrency control (with versioning) is used to provide conversation isolation.

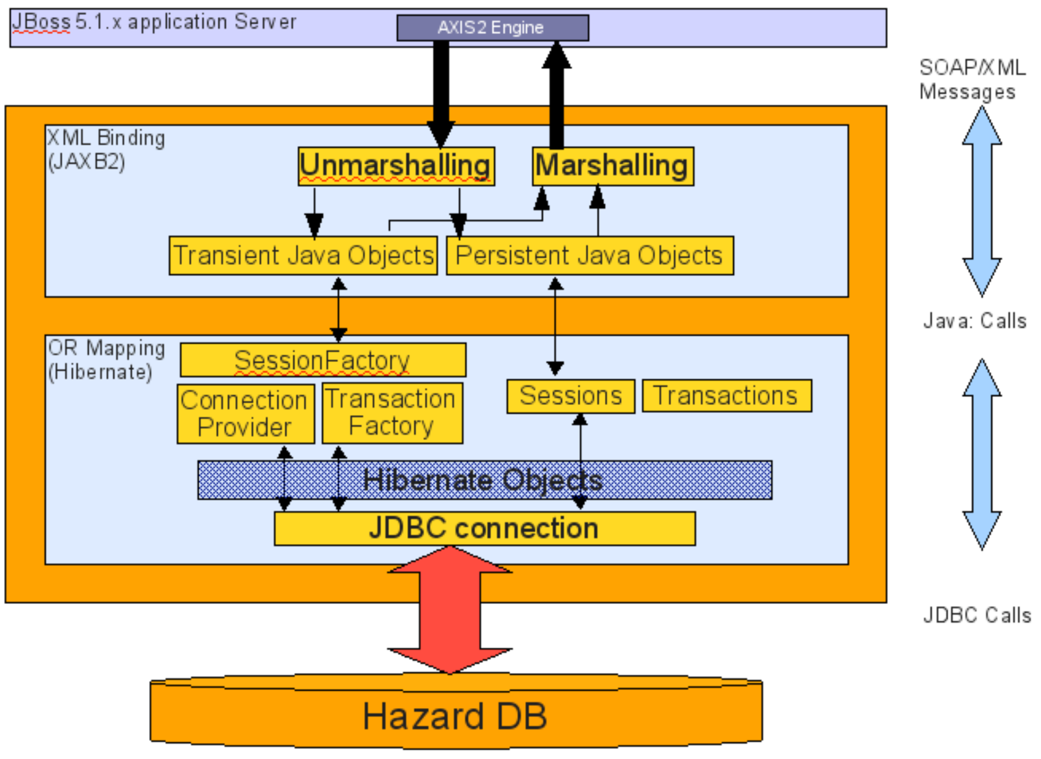


Figure 6.6 A detailed OpenGEM Data Access Layer Architecture

6.3.1 Data Handling with XML and JAXB

The XML (Extensible Markup Language) and related methods are a standard method to define files containing structured data by tagging them with meta data. In the OpenGEM system, the input and output data to the OpenSHA Hazard Engine is provided in XML files.

The GEM Hazard team has defined a structure and all meta data that is allowed to be contained by XML files that are either data input to or data output from OpenSHA Hazard Engine, such as logic trees, seismic sources and hazard curves. That definition is called shaML (Seismic Hazard Assessment Markup Language) and is specified in the GEM Technical report 2010-2 (Pagani et. al).

The shaML defines the type and the structure of the data to describe (e.g. logic trees, seismic sources, hazard curves), whereas "shaML files" are instances containing concrete data of e.g. seismic sources, such as coordinates and maximum values of magnitudes. To sum up, “shaML files” can be referred as a subset of “XML files”. shaML files contain data on seismic hazard in a XML format.

Data Binding and Conversion

JAXB (Java Architecture for XML Binding) is a standard software tool by SUN. In the OpenGEM system, JAXB 2.0 is used to translate the shaML format into Java code and to read, to write and to validate shaML files.

The JAXB's binding facility is used to generate Java code portions (classes) that are the exact representations of the structures defined by the shaML (e.g. seismic sources). In a running program these data representations are called Java objects.

The JAXB's marshalling facility is used to produce a shaML file from these Java objects.

The JAXB's unmarshalling facility is used to read (parse) the data contained by shaML files into Java objects.

The JAXB's validating facility is used to check if a shaML file that is parsed complies with the shaML format. Rules and restrictions defined by shaML are for example the count of elements, the presence of mandatory attributes and that gridded seismic sources contain points as coordinates whereas fault sources are given by fault traces.

Data exchange by shaML files

For the purpose of data exchange, in the OpenGEM system, data is stored and transferred as XML files. Figure 6.7 shows with green arrows and red boxes in what cases the data to and from the calculation flows as XML files.

The blue box shows an example of data that exist in XML structure as Java objects but Here, the data is neither marshalled nor unmarshalled (see section 6.3.1). It is not unmarshalled means that the data is not read from a shaML file but from the database. The shaML file is not marshalled means that is not stored to a file before being processed (for example, sent to a portlet).

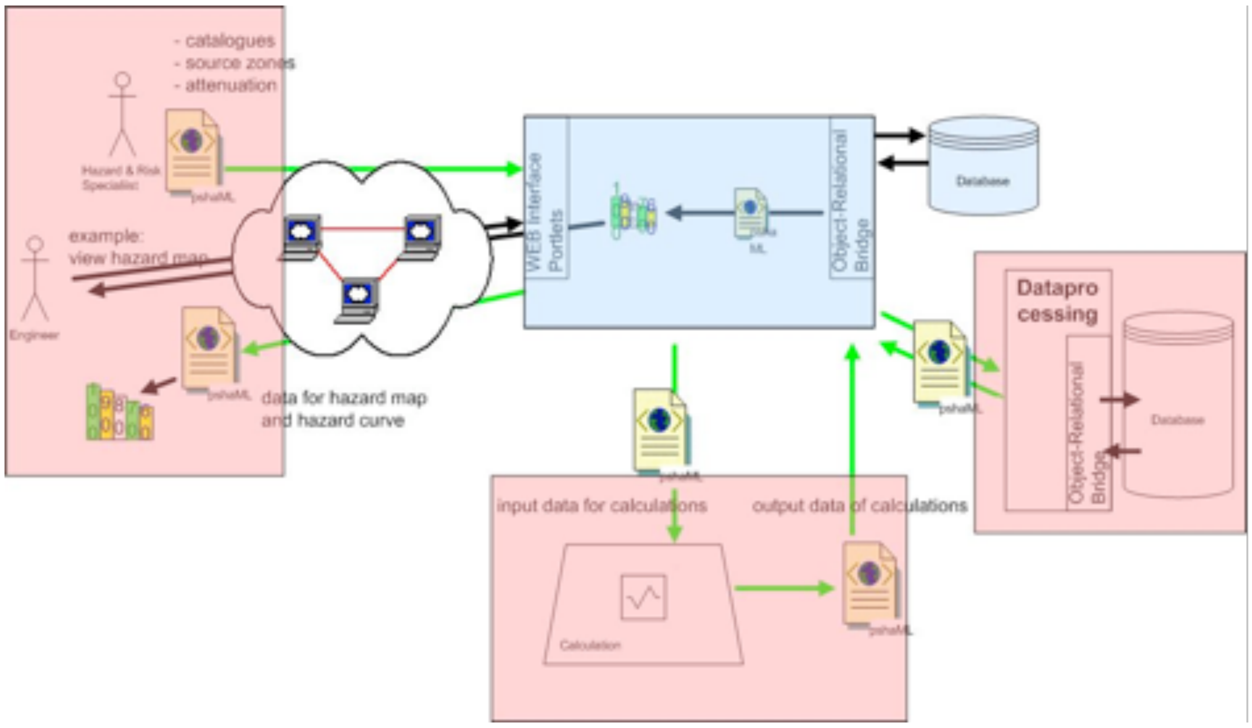


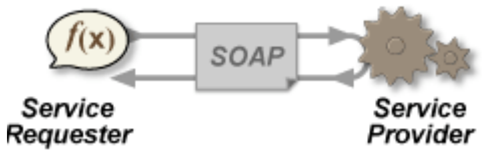
Figure 6.7 Data Flow by XML Files in the OpenGEM System

6.4 Application Design

Various applications – web applications and non-web applications, will be a part of the various components of the OpenGEM system, shown figure 6.1. XML is the widely used data format, also for the *OpenGEM* system. A customized XML format for GEM project is under development called SHAML. Most of the data files that would be transported across different OpenGEM subsystems, OpenGEM components would be in the PSHAML format. XML is the format for exchanging mapped data and for providing

6.4.1 Data Services to/from the OpenGEM system

And webservices is the commonly used data transfer mechanism (i.e., transporting the XML base data files). The illustration below (from Wikipedia, 2009) shows this concept.



SOAP (Simple Object Access Protocol) contains the XML data file along with a header. As an example, the "service requester" and "service provider" in OpenGEM system would be "Data Submission" application and "Data Submission" Portlet. It also implies that web services will help leverage the benefits of SOA. The "Apache Axis2" Engine has all the necessary functionality and follows open standards (of World Wide Consortium).

6.4.2 Communication Between System Components

One basic goal of GEM is to let Engineers and Scientists participate in the OpenGEM system. Another goal is to build a "distributed system" for external partners. These goals need communication of the system's components across computer network boundaries. The data communication in GEM uses the HTTP protocol and the functionality is provided by web services.

The role of web services

Using web services, the application layer implements all functionalities that need to perform data base interaction or application logic. The web services are available to the portal and to the hazard calculation as well.

As an example this illustrates the role of a web service in a communication process: A user of the GEM system chooses to view the hazard curves of a model to specify, he chooses the necessary parameters on the portal site. A web service transmits the parameter to the application layer which performs the related work (e.g. data base interaction), creates an answer and sends it to the portal.

The web service's answer can consist of a shaML file which can be visualized, stored or forwarded.

Distribution

The Hazard sub-system can be roughly split into these components: Hazard calculations, Database, Web portal and Application Logic. Each component has its own specific needs. So the calculation has special requirements on CPU power, the data base server needs a fast access to the hard disk. The communication between these components is mainly data exchange. The data input and output of the components is SHAML files. Files can easily be transferred by web services.

There is further functionalities that does not require to transfer files. As an example, the portal shows a list of hazard maps for a specific model. When the portal calls the corresponding service with a model id selected by a user, the service sends back data records with name, damping and probability of exceedance.

These functionality is interfaced by a small number of web services.

Access to external partners

When the shaML format (see section 6.3.1) is published, it is possible for external partners to create input and output data (shaML files). Individuals will be granted access to the system, to submit such data or just to get output data of hazard calculations. The transfer of these shaML files is handled by web services.

6.4.3 Hazard Calculations Engine

This is an complex component, dealing with many sub-module, as listed below

Hazard Core: This forms the basic building unit of the "computational Infrastructure". This sub-module is sufficiently self-contained, that it can be replaced with another similar unit. The "opensha" (along with some tools) would be a part of a "Hazard Core". Figure 6.8, shows one such "Computational Infrastructure Block" (CIB). At the center is the "core engine", which is the PSHA codes, as provided by various individuals/organizations. The core engine is surrounded by two different "pre-calculations" tools at the first layer

Data pre-calculations tools: Deals with pre-calculation done on the input data. An example would smoothing of the events catalog, before doing PSHA calculations.

Non-Data pre-calculations tools: Deals with pre-calculations done without involving input data. And example would be a graphical tool which defines source zones across national boundaries.

The second layer is composed of Parsers which would parse the input data files (in PSHAML format) and convert them to the format acceptable to the "core engine". This would essentially take care programming language mismatches, e.g., Python-to-Java. The third layer, XML processing layer, would ensure that the input/output data files are PSHAML conferment.

The CIB is the major consumer of computational resources : a dedicated Shared Memory Processing OR scheduling a job on a HPCC facility.

The concept of CIB brings the plug-n-play capability to the OpenGEM system.

Hazard Results Viewer: The term "Hazard results" implies two forms of results – Graphical (i.e., Hazard Curves) and Geographical (i.e., Hazard Maps). And, both of these forms become available at the end of Hazard Calculations, and are stored in the OpenGEM Database. The "Hazard Results Viewer" is usually triggered by the "Hazard Core", at the successful completion of a Hazard Calculation. But, it can also be triggered without any Hazard Calculation being performed, this happens when the user does not have access rights to trigger a hazard calculation.

Graphical form involves obtaining the data required for plotting (from the database) and using a plotting tool (such as GMT tools). The resultant graphical plot is transmitted (over the network) to the user's portlet window in a PNG (Portable Network Graphics) format.

In a Geographical form, the user request is processed with the help from Mapserver sub-module. The Results data are read from the database and transmitted to the Mapserver application residing in the opengem Mapserver. The Mapserver application then adds the necessary background layers for better visual content and stores the resultant graphic file (.png format) in a predetermined location. This location URL is send to the Hazard Results Viewer for onward transmission to the Hazard Viewer Portlet.

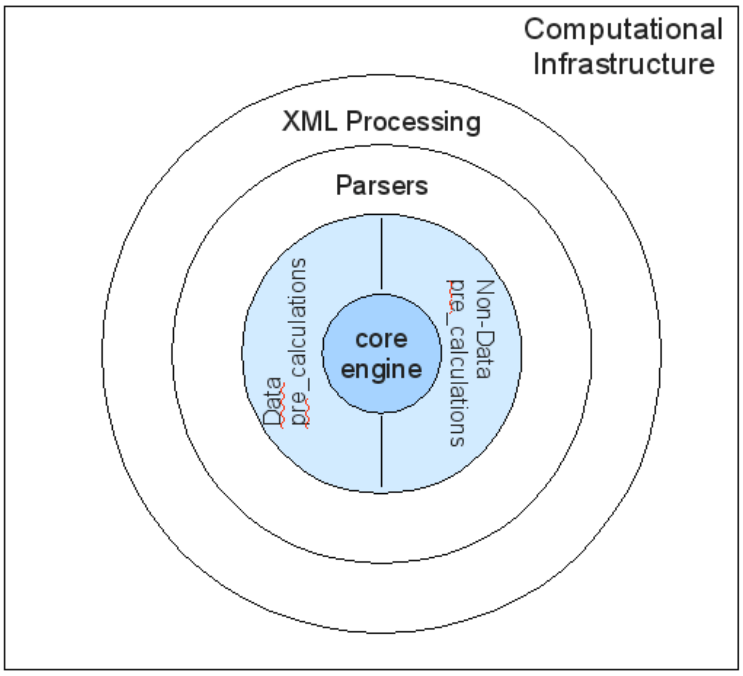


Figure 6.8 A Computational Infrastructure Block

Past Calculations Viewer: A user of the system is interested in tracking calculations performed at a certain point of time in the past. This requires that all hazard calculations performed in the OpenGEM system, be tracked. There are possibilities, when the output data are huge, that one uses/loses large amounts of storage space. Hence, it is decided that only metadata for EACH hazard calculation is stored in the database. This metadata would be sufficient for the user to repeat his earlier calculations.

This module would find and transmit the metadata matching to the user search request. The response would be received by the Hazard Calculations Portlet, the user can then rerun the calculation, which would then be handled by the "Hazard Core" module.

Computational Time Estimation: In a strict sense, computational time are dependent on the hardware, environment and is considered an complex issue to estimate. However, in OpenGEM system, the estimation is used to make the decision, should the computation be done on a HPCC resource or on a SMP resource. In the initial stage, it could be an approximation based on the past calculations and test calculations done on available hardware/environment. In an extreme case, it could also be manual- user decides on prior experience, how/where to do the calculations.

Risk Calculations Engine

Not yet integrated.

Validations Calculations Engine

No design yet

Data Submission

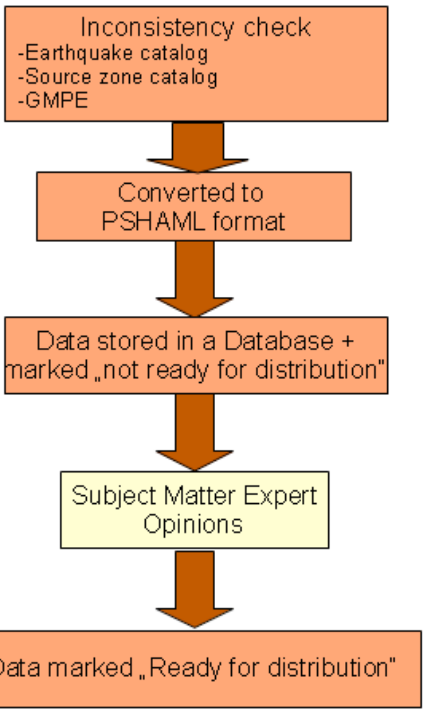

Inconsistency check
-Earthquake catalog
-Source zone catalog
-GMPE
Converted to PSHAML format
Data stored in a Database + marked „not ready for distribution"
Subject Matter Expert Opinions
Data marked „Ready for distribution"

Figure 6.9 Data Submission Flowchart

Figure 6.10 shows the flowchart to be followed when Data/Catalogs enters the OpenGEM system. The inconsistency check could be semi-automatic to begin with, eventually becoming automatic. The "Subject Matter Expert" opinion is a manual procedure and can enforce a four-eye principle.

A Workflow Engine, Apache ODE (Orchestration Director Engine) executes business processes written following the WS-BPEL standard. It talks to web services, sending and receiving messages.

This workflow engine GUI would be available inside the "Data Submission Portlet".

System Administration

This section deals with administration/generation of

Users: The user details (including access rights) will be stored as openLDAP directories, maybe in a separate database. Role Based Access Control (RBAC) will be implemented using openLDAP directories.

Data Services: These services provide data (e.g., different catalogs) (in XML format) to user in the form of web-services. These could also include WMS services from the mapserver. The user will have to subscribe and "pull" these data services. The opengem system will not pro-actively "push" these data services to the users.

RSS feed: These are again "pull" services. The information provided in these feeds are of general nature: changes to usage policies, changes to static content in the opengem website

Each of the above functionality would be implemented using Enterprise Java Beans 3.x technologies.

HPCC Job Manager

This application manages the computational job processing and scheduling to a "High Performance Computing and Communication" infrastructure. One of the well-known product is "Condor".

"Condor is a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, Condor provides a job queuing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to Condor, Condor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion." - http://www.cs.wisc.edu/condor/

The Condor GUI would be available in a separate to-be-built HPCC Portlet

Job Scheduler: Whenever a hazard calculation is broken into smaller job and submitted to a HPCC infrastructure, the dependencies between these smaller jobs are not handled by the Condor scheduler. Hence, a Meta-Scheduler called "DAGman" (a part of the Condor tools) is to be additionally used.

Job Information Agent: This acts as a translator of "Condor jobs" to "opengem job ID".

OpenGEM Interactive Mapping Component

The system provides a mapping interface using a UMN mapserver installation and Openlayers technology (OpenLayers is a project of the Open Source Geospatial Foundation.). The hazard and risk results/data will be provided to GIS clients (such as ESRI ArcGIS), using openGIS standardized WMS (web map services) and WFS (web feature services). The opengem portlets are consumers of the WMS and WFS service.

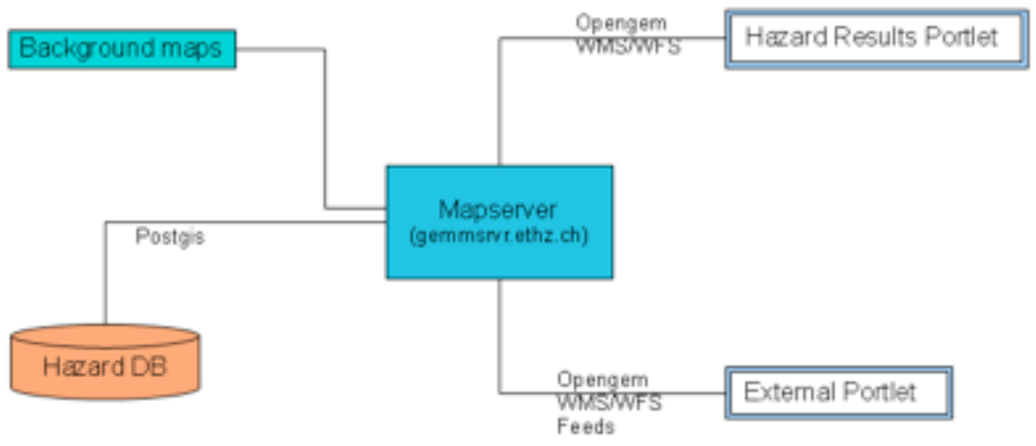


Figure 6.11 Mapserver Application

OpenGEM contains an interactive mapping application that has been set up using an open source software stack. On the server side, UMN MapServer (mapserver.org, version 5.6.1) and PostgreSQL/PostGIS (postgis.refractions.net, version 1.5.0) are used. On the client side, the Javascript library OpenLayers (www.openlayers.org, version 2.8) is used. It provides an interactive map window, including controls for typical web mapping functionality like pan, zoom, layer selection, and overview map. The map window can be embedded in portlets that require the display of maps, like the hazard curve viewer and hazard map viewer portlets. The geospatial data sets to be portrayed on a map are stored in the OpenGEM database. For that purpose, the PostGIS data type POINT is used. This allows the MapServer application to directly access the database as a data source for a vector layer. If the character of the data set is that of a spatial coverage that has been sampled at a set of grid points, the spatial extent of the grid cells is computed on-the-fly using PostGIS functions. The following example is the DATA directive of a MapServer configuration file for the display of hazard maps.

```
DATA "the_geom FROM (
    SELECT gp.gpid AS gpid,
        ST_Expand( gp.gppgpoint, hzm.hmapgridsize*0.5 ) AS the_geom,
        hpv.hpvalue AS hpvalue
      FROM geopoint gp
      INNER JOIN hazardpointvalue hpv ON ( hpv.gpid = gp.gpid )
      INNER JOIN hazardmap hzm ON ( hzm.hcid = hpv.hcid )
      INNER JOIN hazardcalculation hzc ON ( hzc.hcid = hzm.hcid )
      WHERE ( hzm.hmapid = %HMAPID% )
    ) AS subquery
    USING UNIQUE gpid USING SRID=4326"
```

The extent of the grid cells is computed using the PostGIS function ST_Expand(), with the dimension of the grid cells (stored in the database table hazardmap) as an additional paramater. The hazard map to be displayed is selected using the parameter HMAPID, which has to be specified in the MapServer method call. In order to make this database query sufficiently fast, it is important that database indexes are created for columns that are used in JOIN and WHERE clauses.

For OpenGEM, MapServer is used as a Web Map Server am international standard under the auspices of the Open Geospatial Consortium (www.opengeospatial.org). The following example shows the URL of a GetMap() method call that requests a hazard map layer (line breaks bewteen URL parameters have been added for clarity):

```
http://gemmsrvr.ethz.ch/cgi-bin/mapserv?
  MAP=/var/www/mapfile/gemhazard.03.map&
  SERVICE=WMS&
  VERSION=1.1.1&
  REQUEST=GetMap&
  layers=hazardmap&
  STYLES=&
  SRS=EPSG:4326&
  BBOX=-180,-60,180,90&
  WIDTH=1000&
  HEIGHT=417&
  FORMAT=image/jpeg&
  HMAPID=55
```

The OpenLayers Javascript code for a WMS map layer with the same hazard map looks like this:

```
hazard = new OpenLayers.Layer.WMS(
  "GEM Hazard",
  "http://gemmsrvr.ethz.ch/cgi-bin/mapserv?",
  { map: "/var/www/mapfile/gemhazard.03.map",
    layers: "hazardmap", format: 'image/jpeg', srs: 'EPSG:4326',
    transparent: true, hmapid: 55 },
  { isBaseLayer: false, opacity: 0.5 } );
```

In the hazard map viewer portlet, the hazard map data is shown superimposed on relief data that has been derived from a digital elevation model (DEM). In order to create a suitable hillshade data set, base data from the Shuttle Radar Topography Mission (SRTM, http://www2.jpl.nasa.gov/srtm/) with a spatial resolution of 30 arcsec (SRTM30) has been used. The DEM is available in 27 40x50-degree tiles. They have been converted from the original data format to GeoTIFF and merged into one file using the command-line tools gdal_translate and gdal_merge.py from the open-source Geospatial Data Abstraction Library (GDAL, www.gdal.org). The GeoTIFF data format allows internal tiling, which has been enabled using the command line option <-co "TILED=YES">. Subsequently, a hillshade dataset has been computed using the tool “gdaldem” with the option hillshade. In order to serve the relief data efficiently through MapServer, internal overview layers with lower resolution have been computed. For that purpose, the tool “gdaladdo” has been used. Figure 6.11 shows an example of a OpenLayers map window showing a hazard map with the hazard data set superimposed on the relief background, and additional political boundaries. The chosen map projection is equirectangular (plate carée, EPSG code 4326).

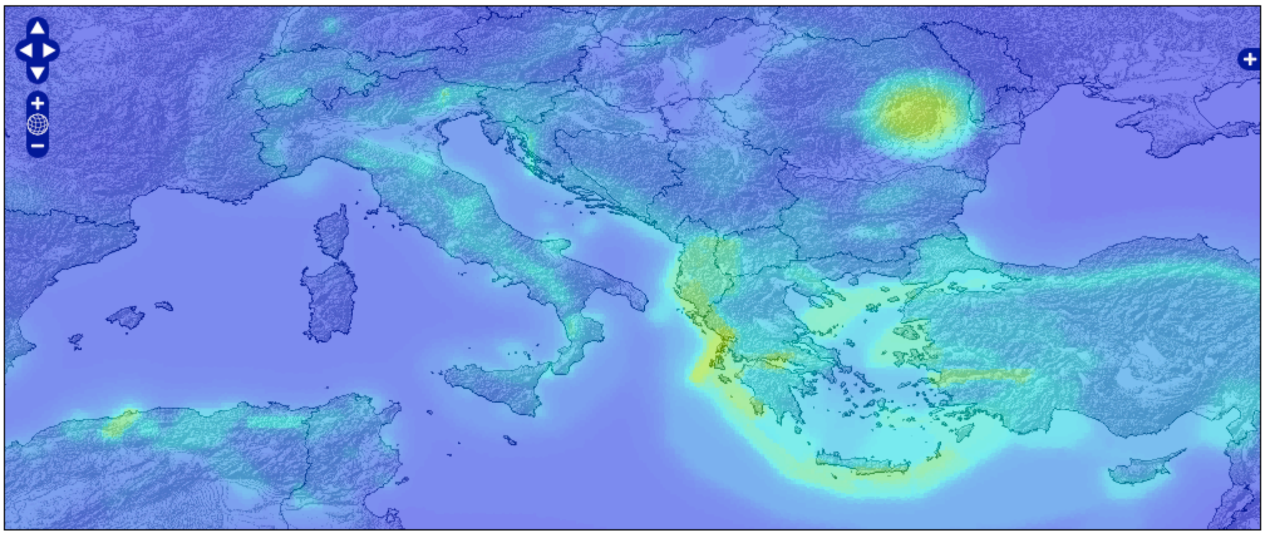


Figure 6.11 Openlayers Map window with Hazard Map

In the hazard curve viewer portlet, an interactive map for site selection is included. It allows to extract the coordinates of a specific point on the map by double-clicking. For this map, the NASA Blue Marble next generation (BMNG, http://earthobservatory.nasa.gov/Features/BlueMarble/) imagery has been used.

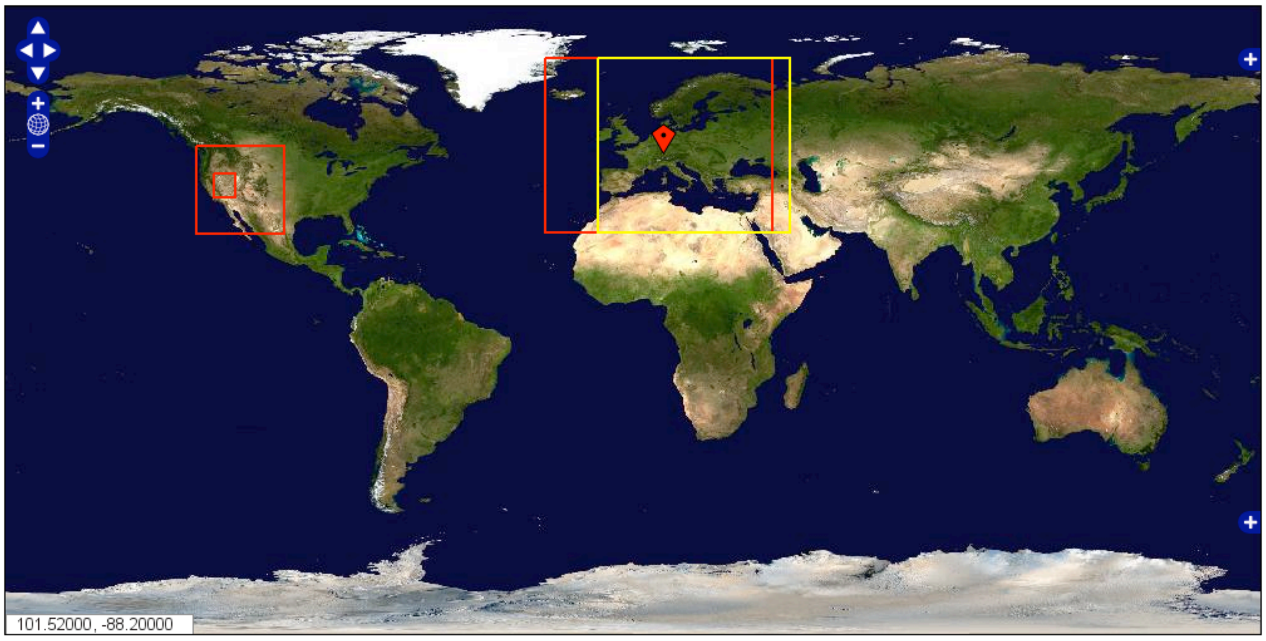


Figure 6.12 Site selection mapping window

BMNG map layers can be requested through a WMS from NASA (http://wms.jpl.nasa.gov/wms.cgi), but for a faster availability a local copy has been installed and is served as a WMS from the OpenGEM map server. For efficient processing by MapServer, internal tiling and overviews have been added to the GeoTIFF images. In addition to the BMNG base layer, the site selection map shows the extent of regions for which hazard computation have been made as polygon outlines. These polygons are stored in the OpenGEM database (tables hazardinputltreemodel and hazardinputbasicmodel) and are served as a vector layer through MapServer. Figure 6.12 shows an example of the site selection mapping window. The map projection is equirectangular.

6.5 Portal design

The Portal framework used is "Jetspeed 2.x" and the portlet container is "Pluto 2.x". Please note this sub-section deals only with the "Presentation tier". The data transfer format used are as follows:

- XML (& its variants) are the data transfer technologies used
- Web Services (using SOAP, WSDL,...) would be used as transfer mechanism
- KML format will be used for the visualization of 3D geospatial data in programs like Google Earth, Worldwind (NASA)
- GML format will be used (WMS & WFS protocols) to visualize GIS maps in program like UMN Mapserver, OpenLayers 2.8.

In the illustration below, there are "Portlet Producers" and "Portlet Consumers". The Portlet Producers are organizations which take-in data from existing databases/repositories and implement a certain functionality/service using portlet technologies. The Portlet Consumer is the GEM Portal, which uses the functionalities to supplement its services. This is a value addition to some special clients/users, who would like their services to be hosted on the OpenGEM portal. This also provides a mechanism to "outsource" the development of certain portlet development.

The OpenGEM Portal will not consume data directly from external software agents (for e.g., as a data web service). All data entering the OpenGEM system has to undergo an Inconsistency/Reality check (see section 6.4.4).

6.5.1 *Corporate Identity*

This refers to the look-n-feel of the Portal (and Portlets). There are two aspects to this – Graphical and non-Graphical. Graphical aspect refers to the colours, corporate logos and patterns. Non-graphical aspects are choice of portlets, placement of portlet(s) on the various areas of the Portal and the portlets themselves

Graphical aspects are combined into a template, more precisely a Velocity template. The idea is to have a "GEM template", but also the option for users to customize the Portal using the "available" templates.

Non-graphical aspect is achieved using concept of portlet layouts. This aspect could be "locked/setup" by the system designer based on the user profiles.

The Portlets represent the start of the chain-of-processing steps for each functionality (sometimes usecases). Many of the portlets are shown in the following sub-sections.

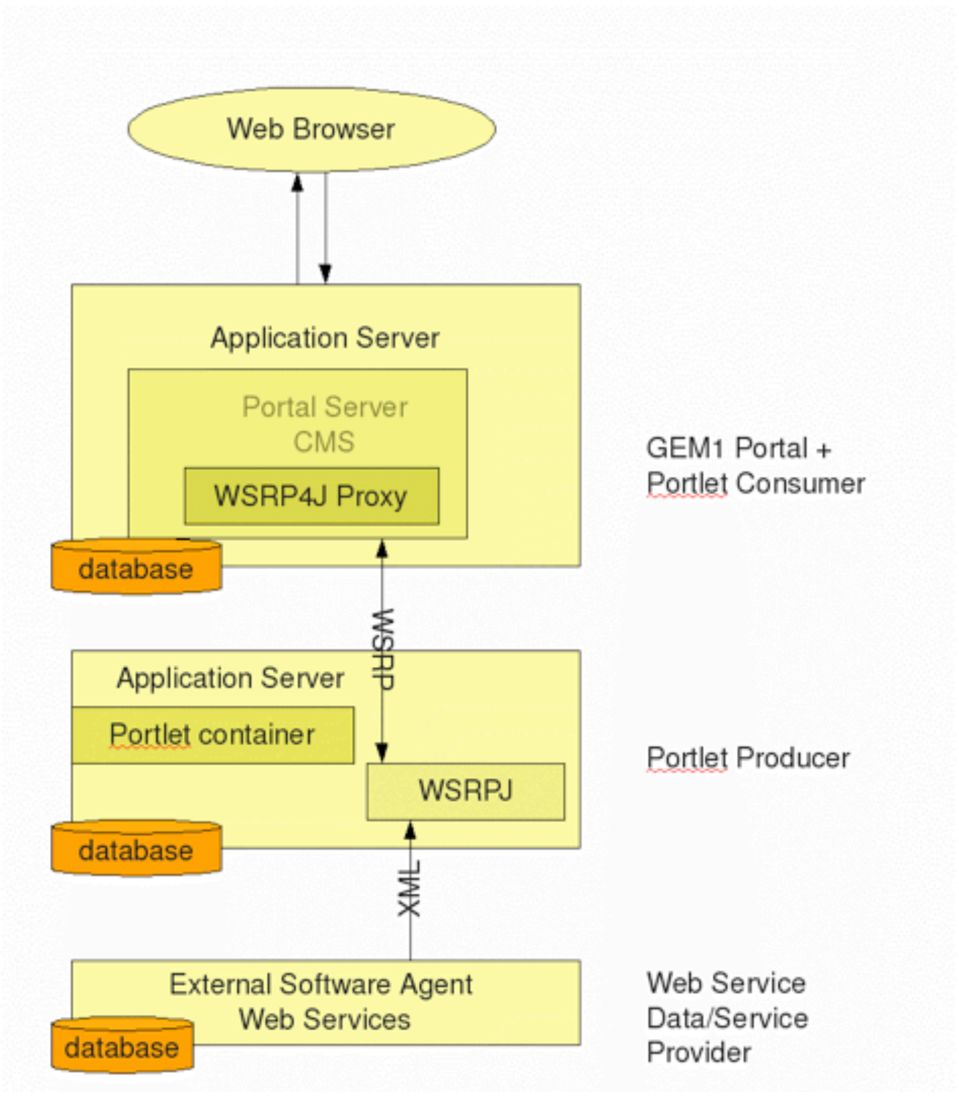


Figure 6.13 The OpenGEM Portal.

6.5.2 *Hazard Calculation Portlet*

The schematic below shows how the simple hazard calculation steps spans various tiers.

Figure 6.14 shows how this portlet window would look. This Portlet contains the functionality described in the section - "Usecase: Hazard Calculations".

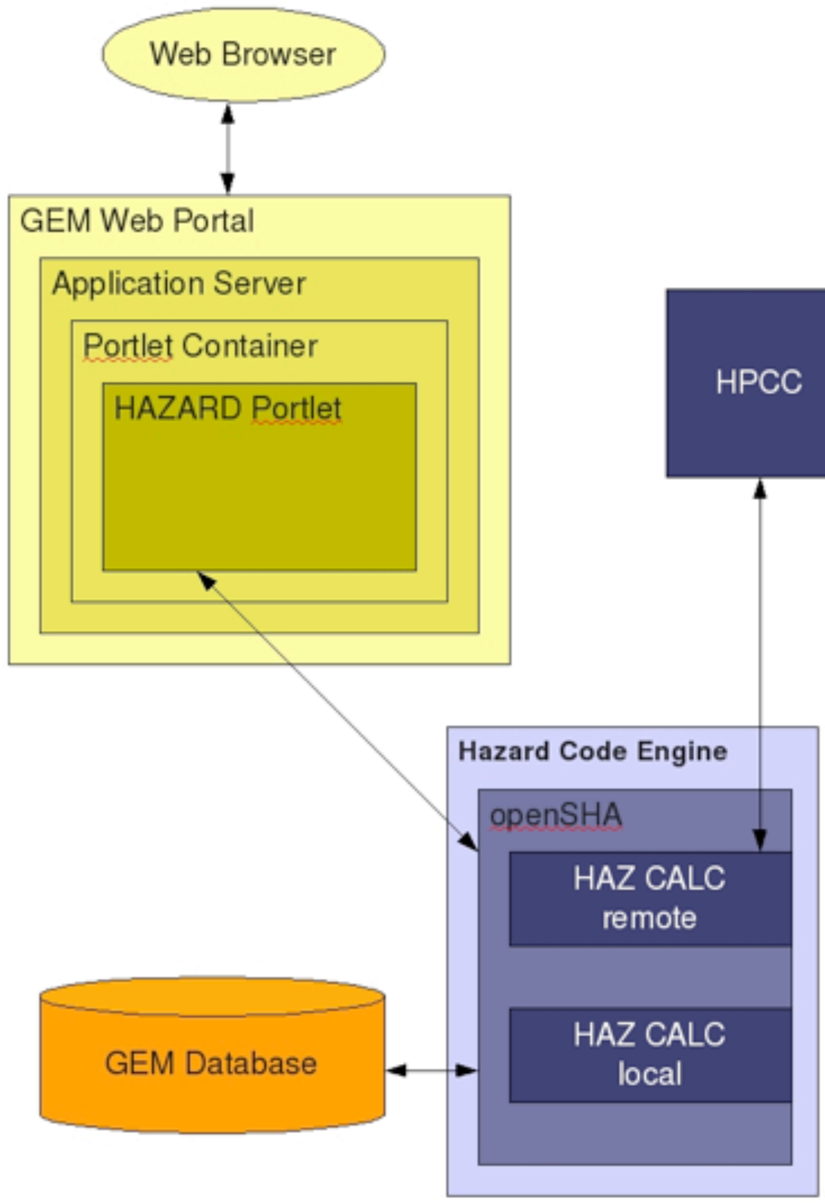


Figure 6.14 A simple hazard calculation

Layout

It contains four tabbed regions, labelled as Region Selection, Model selection, Calculation configuration and Analysis. Each of these tabbed region is described in details here. Each Tab contains several "interaction boxes" (with _box suffix), which contain graphical elements like buttons, text area, etc.

Region Selection Tab

Region Selection	
regionSelection_box (box-1)	quadrilateralFreeshape_box (box-2)
	regionEndpoints_box (box-3)
	_box (box-0)

box-1: Site selection

A first option presents a small map with the usual options zoom in, zoom out, drag (Openlayers based).

Technically, the WMS displayed is hosted by ETH. Its URL is defined in the web.xml configuration file of the Web application.

The user clicks on the map to select a location. Two fields (input text): x, y (lat/long) are automatically filled with the values clicked. The user can optionally edit the fields to modify the values then he submits the request.

In the same box, the user can choose between "bedrock" (who is the value by default) or "site condition". This information would be available as a web service.

box2: select model calculation

The model calculation selection box presents three options:

1) displays the mean (one curve who is a synthesis of the model). This option is chosen by default;
2) displays the mean and the fractiles (several curves that display the (?) will then be displayed);
3) choose a particular path.

If the user selects the option 3), the system displays the list of possible values:

3a) The client displays the list of the distinct names of rupture forecast available for this point

3b) The user chooses a rupture forecast

3c) According to this first choice, the client displays the list of distinct names of source models existing for this point, for this given rupture forecast

3d) The user chooses a rupture forecast

3e) According to this second choice, the client displays the list of distinct values of GMPE existing for this point, for these given rupture forecast and source model

3f) The user chooses a value for GMPE

This set (rupture forecast, source model, GMPE) is particular path of the logical tree.

Technically, to build this interface the client requests a SOAP web service hosted by ETH:

- ⇨ input: (x,y) : site (Finally chosen/modified by the user in BOX0) and Site condition (“bedrock” or “other site condition”)
- ⇨ ouput: the list of (a,b,c) available for this site. Where
  - a is “rupture forecast” (name of)
  - b is “source model” (name of)
  - c is GMPE (numerical value)

<u>box-3: results sub-selection: what we want to display</u>

Model refers for this particular point (x,y) to data points with the following properties:

- period (~0…10sec = 1/frequency): x axis of the graph. The user does not select that one, it is always the x axis. The client will determinate the scale to use from the data.
- several spectral motion description: define the unit of the y axis of the graph
  - spectral acceleration (default choice)
  - spectral velocity
  - spectral displacement

1) selection of the unit of the y axis: the client displays the list of names of spectral motion descriptions. The user selects one of the values.

Technically, this list is fixed in the client. It is configured in the web.xml configuration file of the web application. *This list should be stored in the GUI side.*

2) selection of the y axis to display: the client displays the available probability of Exceedances for the given spectral motion selected. The user selects one.

Technically, the client requests a SOAP web service to obtain the list of probability of Exceedances:

- ⇨ Input: model (a,b,c) or “mean” or “mean and fractiles”; and site (x,y); and Site Condition (“bedrock” or “other site condition”); and y axis (a term which describes the spectral motion description)
- ⇨ Output: a list a numerical values (2%, 5%….)

The client displays this list which is the available probabilities of Exceedances.

The curve is now defined.

<u>box-4: the curve</u>

The client displays the curve defined by BOX1 (site), BOX2 (model), BOX3 (information to display).

Technically, the client requests a SOAP web service:

- ⇨ Input: the model (a,b,c) or “mean” or “mean and fractiles”; the site (x,y); Site condition (“bedrock” or “other site condition”); the y axis (a term); the probability (numerical values).
- ⇨ Output: the data (*the exact XML schema (SHAML OUTPUT) is not yet defined*)

box-5: other displays/export of the data

This part needs to be defined in more detail.

a. display the data
b. display the parameters used
c. button to:
   - download the XML (~SHAML)
   - export the data in csv. for local saving
   - print a report: a free text area allows the user to enter a comment. The PDF report contains the image (curves), the data, the definition of the model calculation, the parameters (2), and the comment. Almost half of the page is the curve, bottom: left is data, right is notes.
   - save the curve in png without the axis (only the curve): it's "peel off". The legend must be inside the curve area.
   - Log the axis (x/y): logarithm. So changing the axis is possible.

box-0: choice of another curve

Once a curve is defined and displayed, the user can choose another curve (using box1, box2, box3) and display it on the same graph window (box4). In the application, there is not any limitation on the number of such curves displayed on the same graph.

In this box, the user can also delete a curve, hide a curve, or update the parameters defining a curve.

Hazard Curve Tab:

Hazard Curve		
curve_box (box-4)	site_box (box-1)	modelSelection_box (box-2)
dataLocal_box (box-5)		
	curveDisplay_box (box-3)	multipleCurve_box (box-0)

box-1: Site selection

A first option presents a small map with the usual options zoom in, zoom out, drag. (Openlayers based)

Technically, the WMS displayed is hosted by ETH. Its url is defined in the web.xml configuration file of the Web application.

The user clicks on the map to select a location. Two fields (input text): x, y (lat/long) are automatically filled with the values clicked. The user can optionally edit the fields to modify the values then he submits the request.

In the same box, the user can choose between “bedrock” (who is the value by default) or “site condition”. This information would be available as a web service.

<u>box-2: select model calculation</u>

The model calculation selection box presents three options:

1. displays the mean (one curve who is a synthesis of the model). This option is chosen by default;
2. displays the mean and the fractiles (several curves that display the (?) will then be displayed);
3. choose a particular path.

If the user selects the option 3), the system displays the list of possible values:

3a) The client displays the list of the distinct names of rupture forecast available for this point

3b) The user chooses a rupture forecast

3c) According to this first choice, the client displays the list of distinct names of source models existing for this point, for this given rupture forecast

3d) The user chooses a rupture forecast

3e) According to this second choice, the client displays the list of distinct values of GMPE existing for this point, for these given rupture forecast and source model

3f) The user chooses a value for GMPE

This set (rupture forecast, source model, GMPE) is particular path of the logical tree.

Technically, to build this interface the client requests a SOAP web service hosted by ETH:

input: (x,y) : site (chosen by the user in BOX0) and Site Condition (“bedrock” or “other site condition”)

output: the list of (a,b,c) available for this site. Where

a is “rupture forecast” (name of)

b is “source model” (name of)

c is GMPE (numerical value)

<u>box-3: results sub-selection: what we want to display</u>

Model refers for this particular point (x,y) to data points with the following properties:

hazard measure (ground motion): x axis of the graph. The user selects one of many. Values to be selected from: “PGA” (default value), “PGV”, “intensity”, …

probability of Exceedance : y axis. The user does not select that one, it is always the y axis. The client will determinate the “scale” from the data.

1) selection of the unit of the x axis (ground motion): the client displays the list of names of ground motion description. The user selects one of the values.

Technically, this list is fixed in the client. It is configured in the web.xml configuration file of the web application. *This list should be stored in the GUI side.*

selection of the x axis to display: the client displays the available numerical values (50, 1000) for the given ground motion selected. The use selects one value in this list.

Technically, the client requests a SOAP web service to obtain the list of periods:

- ⇨ Input: model (a,b,c) or "mean" or "mean and fractiles"; and site (x,y); and Site Condition ("bedrock" or "other site condition"); and x axis (a term which describes the ground motion)
- ⇨ Output: a list a numerical values (50, 1000….)

The client displays this list.

The curve is now defined.

box-4: the curve

The client displays the curve defined by BOX1 (site), BOX2 (model), BOX3 (information to display).

Technically, the client requests a SOAP web service:

- ⇨ Input: the model (a,b,c) or "mean" or "mean and fractiles"; the site (x,y); Site Condition ("bedrock" or "other site condition"); the x axis (a term); the period (a numerical value)
- ⇨ Output: the data (*the exact XML schema/SHAML-OUTPUT is not yet defined*)

box-5: other displays/export of the data

This part must be defined in more detailed. It will be the same that for Hazard spectra.

box-0: choice of another curve

As for Hazard spectra tab, the user can choose another curve.

Interactions

All the various interaction to acheive the functionality are shown in Illustration-13 : User interactions with the portlet, Web Services Interaction and Interaction between the portlets (or within individual portlets).

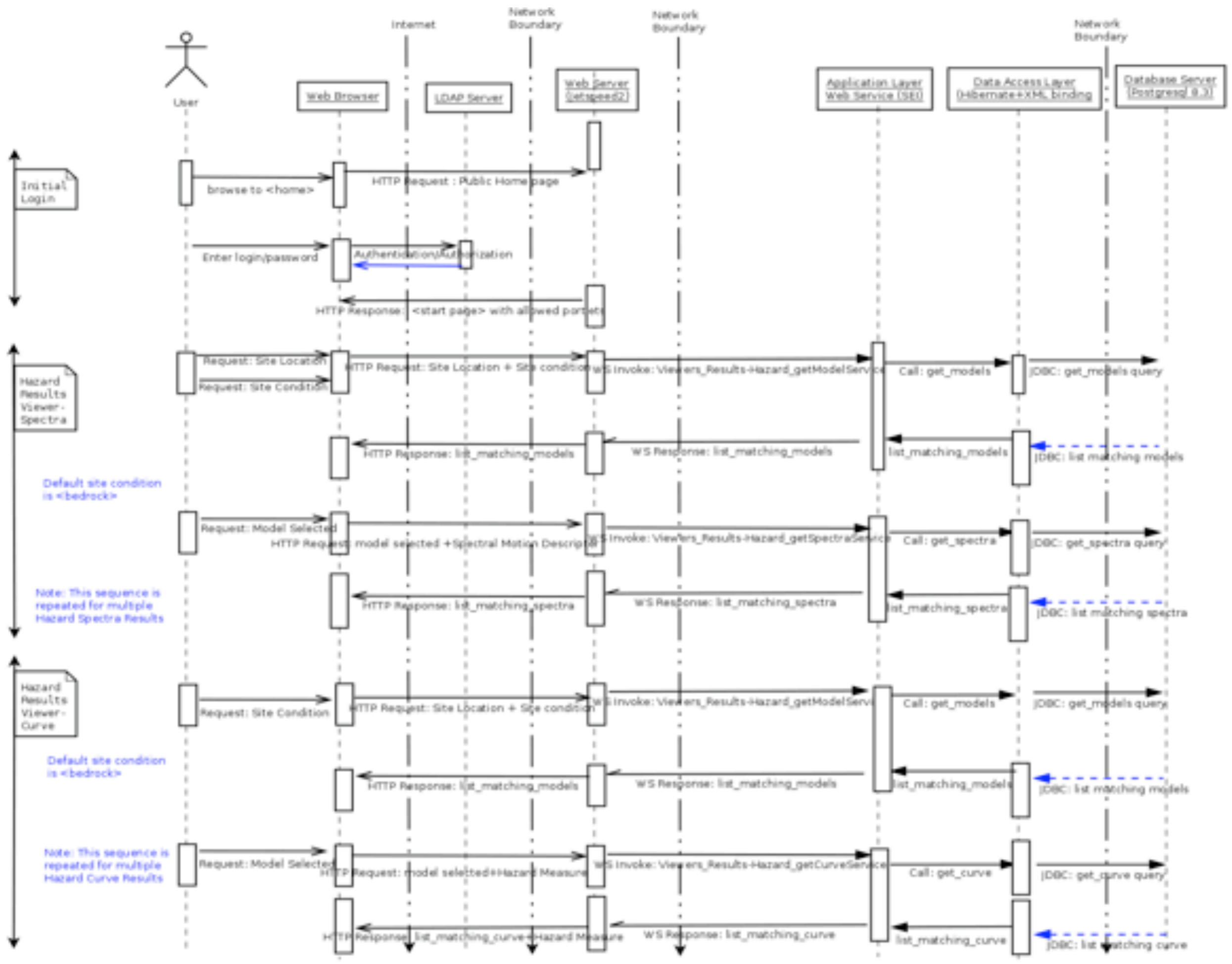

Internet
Network Boundary
Network Boundary
Network Boundary
User
Web Browser
LDAP Server
Web Server (Jetspeed2)
Application Layer Web Service (SEI)
Data Access Layer (Hibernate+XML binding
Database Server (Postgresql 8.3)
Initial Login
browse to <home>
HTTP Request : Public Home page
Enter login/password
Authentication/Authorization
HTTP Response: <start page> with allowed portlets
Hazard Results Viewer-Spectra
Request: Site Location
Request: Site Condition
HTTP Request: Site Location + Site condition
WS Invoke: Viewers_Results-Hazard_getModelService
Call: get_models
JDBC: get_models query
HTTP Response: list_matching_models
WS Response: list_matching_models
list_matching_models
JDBC: list matching models
Default site condition is <bedrock>
Request: Model Selected
HTTP Request: model selected +Spectral Motion Description
WS Invoke: Viewers_Results-Hazard_getSpectraService
Call: get_spectra
JDBC: get_spectra query
HTTP Response: list_matching_spectra
WS Response: list_matching_spectra
list_matching_spectra
JDBC: list matching spectra
Note: This sequence is repeated for multiple Hazard Spectra Results
Hazard Results Viewer-Curve
Request: Site Condition
HTTP Request: Site Location + Site condition
WS Invoke: Viewers_Results-Hazard_getModelService
Call: get_models
JDBC: get_models query
Default site condition is <bedrock>
HTTP Response: list_matching_models
WS Response: list_matching_models
list_matching_models
JDBC: list matching models
Note: This sequence is repeated for multiple Hazard Curve Results
Request: Model Selected
HTTP Request: model selected+Hazard Measure
WS Invoke: Viewers_Results-Hazard_getCurveService
Call: get_curve
JDBC: get_curve query
HTTP Response: list_matching_curve+Hazard Measure
WS Response: list_matching_curve
list_matching_curve
JDBC: list matching curve

Figure 6.16 The sequence diagram for the Hazard Results Viewer Portlet

Web Services Involved

The following web service will be used to obtain the necessary result sets from the database.

Viewers_Results-Hazard_getModelService	
Input Message	Float Site_lat Float site_long String Site_Condition
Processing	Returns a list of model_name corresponding to the Input provided, from the Hazard DB, The list contains all the model name categorized to Rupture Forecast, Source Model and GMPE. The selected ResultSet is “converted” to SHAML-OUTPUT.
Output Message	Returns a list of models (string model_name)

Viewers_Results-Hazard_getSpectraService	
Input Message	String model_name Float Site_lat Float site_long Float Spectral_Motion_Descripter (Displacement(default), Velocity and Acceleration)
Processing	Makes a matching selection from Hazard DB (information contained in table-HAZARDMAP). The selected ResultSet is “converted” to SHAML-OUTPUT.
Output Message	Returns a list of “probability of exceedances”

Viewers_Results-Hazard_getCurveService	
Input Message	String model_name Float Site_lat Float site_long Float Hazard Measure (PGA, PGV, Intensity)
Processing	Makes a matching selection from Hazard DB (information contained in table-HAZARDMAP). The selected ResultSet is “converted” to SHAML-OUTPUT.
Output Message	Returns a list of “probability of exceedances”

Technologies Used

This portlet will based on JSR-286 standards.The various GUI would be developed using AJAX. The mapping need would be met by the use of the “Openlayers” (a Javascript based libraries). The WMS is provided using UMN mapserver.

Input/Output

All the input and output data will be represented using GEM proprietary XML-based format called SHAML-INPUT and SHAML-OUTPUT format.

Figure 6.17 shows how this portlet could look like. Openlayers offers several controls on the mapped results for e.g., user has the option to choose to view one or more layers, at the each time.

Hazard Results Portlet

The schematic below shows how the simple hazard calculation steps spans various Tiers.

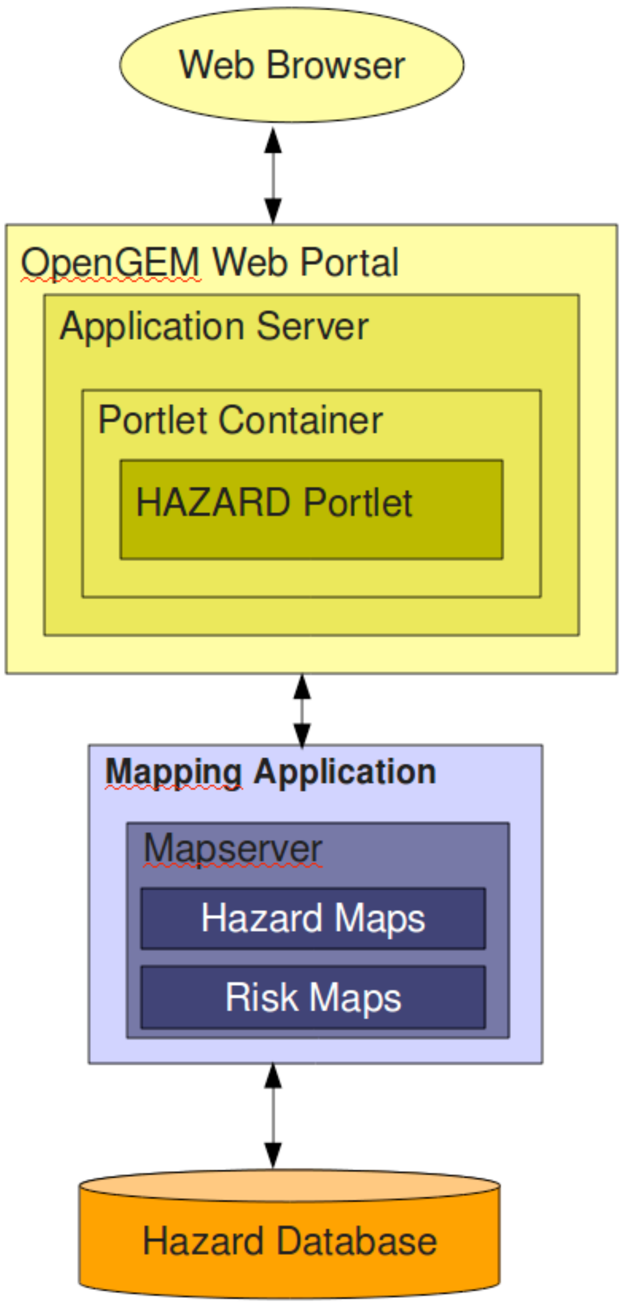


Figure 6.17 A Hazard Results – Hazard Maps Viewer

The figure shows how this portlet window would look. This Portlet contains the functionality described in the section - "Usecase: View Hazard Results".

Layout

It contains one tabbed regions: Hazard Results. This tabbed Region contains several “interaction boxes” (with _box suffix), which contain graphical elements like buttons, text area, etc.

Hazard Results Tab:

Selection_box (box-1)	Map_box (box-2)	
	Output_box (box-3)	Data_box (box-4)

box-1: Selection

This option presents a global map with the usual options of zoom in, zoom out, drag (Openlayers based).

The user clicks on the map to select a location. Two fields (input text): x, y (lat/long) are automatically filled with the values clicked. The user can optionally edit the fields to modify the values then he submits the request. The user also has the option of selecting an area on the maps, Upper Right and Lower Left corners.

Based on the site/area selected, the user is presented with a list of available/applicable Hazard model.

The user chooses one model of interest, from the list, then the user is presented with a matrix of corresponding IMT parameters in a table. The user can now make his choice corresponding to his/her interest.

In the same box, the user can choose between “bedrock” (who is the value by default) or “site condition”. The user may now click on the “display maps” button. And, the corresponding Hazard Maps are displayed in box-2

box2: Map Display

The various mapping layers are discussed in section 6.4.8.

box-3: Map Output

Various export and printing options are available, once the maps are generated

download the maps as a graphic file (*.jpg, *.png, *.gif)

export the map in KML format, locally

print the maps, as *.pdf.

box-4: Data Output

All the user interactions are “recorded” in this box. The recording consists of:

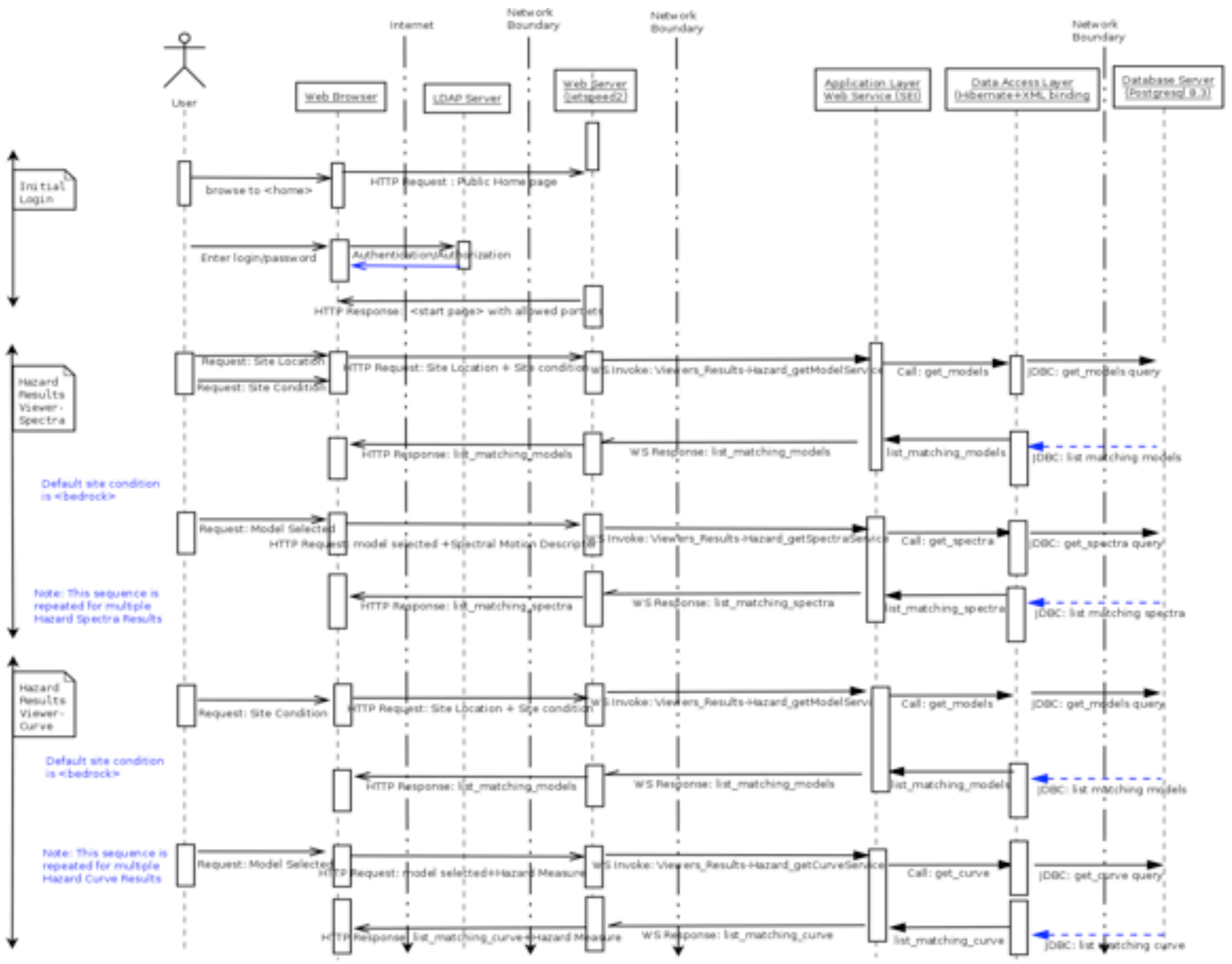
Display the data

Display the parameters used in box-1

Possibility to generate a report, as a .pdf. The format/content is predetermined by the system administrators.

Interactions

All the various interaction to achieve the functionality are shown in figure 6.17: User interactions with the portlet, Web Services Interaction and Interaction between the portlets (or within individual portlets).



User
Web Browser
LDAP Server
Web Server (jetspeed2)
Application Layer Web Service (SEI)
Data Access Layer (Hibernate+XML binding)
Database Server (Postgresql 8.3)
Internet
Network Boundary
Network Boundary
Network Boundary
Initial Login
Hazard Results Viewer-Spectra
Hazard Results Viewer-Curve
browse to <home>
HTTP Request : Public Home page
Enter login/password
Authentication/Authorization
HTTP Response : <start page> with allowed portlets
Request: Site Location
Request: Site Condition
HTTP Request: Site Location + Site condition
WS Invoke: Viewers_Results-Hazard_getModelService
Call: get_models
JDBC: get_models query
HTTP Response: list_matching_models
WS Response: list_matching_models
list_matching_models
JDBC: list matching models
Default site condition is <bedrock>
Request: Model Selected
HTTP Request: model selected +Spectral Motion Description
WS Invoke: Viewers_Results-Hazard_getSpectraService
Call: get_spectra
JDBC: get_spectra query
Note: This sequence is repeated for multiple Hazard Spectra Results
HTTP Response: list_matching_spectra
WS Response: list_matching_spectra
list_matching_spectra
JDBC: list matching spectra
Request: Site Condition
HTTP Request: Site Location + Site condition
WS Invoke: Viewers_Results-Hazard_getModelService
Call: get_models
JDBC: get_models query
Default site condition is <bedrock>
HTTP Response: list_matching_models
WS Response: list_matching_models
list_matching_models
JDBC: list matching models
Note: This sequence is repeated for multiple Hazard Curve Results
Request: Model Selected
HTTP Request: model selected+Hazard Measure
WS Invoke: Viewers_Results-Hazard_getCurveService
Call: get_curve
JDBC: get_curve query
HTTP Response: list_matching_curve+Hazard Measure
WS Response: list_matching_curve
list_matching_curve
JDBC: list matching curve

Figure 6.17 The sequence diagram for the Hazard Results Viewer Portlet

Web Services Involved

The following web service will be used to obtain the necessary result sets from the database.

Viewers_Results-Hazard_getHazardMapService	
Input Message	Integer ModelID
Processing	Returns a list of model_name corresponding to the Input provided, from the Hazard DB, The list contains all the model name categorized to IMT parameters
Output Message	Returns a list of the following

	Hazard Model Short Name Hazard Map Short Name IMT Probability of Exceedance Time Span Damping Hazard Map ID

Viewers_Results-Hazard_getModelService	
Input Message	Float Site_lat Float site_long String Site_Condition
Processing	Returns a list of model_name corresponding to the Input provided, from the Hazard DB, The list contains all the model name categorized to Rupture Forecast, Source Model and GMPE. The selected ResultSet is "converted" to SHAML-OUTPUT.
Output Message	Returns a list of models (string model_name)

Technologies Used

This portlet will based on JSR-286 standards.The various GUI would be developed using AJAX. The mapping need would be met by the use of the "Openlayers" (a Javascript based libraries) The WMS is provided using UMN mapserver.

Input/Output

All the input and output data will be represented using GEM proprietary XML-based format called SHAML-INPUT and SHAML-OUTPUT format.

Openlayers offers several controls on the mapped results for e.g., user has the option to choose to view one or more layers, at the each time.

6.5.3 Risk Calculations Portlet

To be decided. See also Technical Report 5; GEM1 Seismic Risk Report: Part 2.

6.5.4 Risk Results Portlet

To be decided. See also Technical Report 5; GEM1 Seismic Risk Report: Part 2.

6.5.5 System Administrator Portlet

This is an Admin Portlet, which implies access to only the system administrator. As mentioned in section 6.4.5, important system administration will occur through this portlet.

6.5.6 HPCC Portlet

This portlet allows to set up the parameters of cluster computing tool suite as e.g. Condor (tested in cooperation with USGS) in order to have a pre-configured hazard or risk calculation executed on a cluster facility. This GUI should enable the user to issue all the command that a Condor CLI would do.

APPENDIX A Portlet Frontend Specifications

The following text is on the portlet frontend specifications, as provided by the SHARE deliverable D 6.2 (Portal design specification document) – *for the SHARE-portal*, sections 5.3 – 5.6 (By Wössner et al., unpublished). They are developed to be in phase with the technical requirements for the GEM1 portal-mounted applications, however, the layout of the mockups is SHARE-specific.

A.1 Usecase: Hazard Map Portlet (HMP)

The HMP allows the user to access pre-computed result that display information on a particular hazard intensity measure type (IMT). The user's area of interest a region, information on a particular site should be gained through the HCP. The user is interested in obtaining information about various relevant hazard results. The model selection procedure follows an internal logic which turns on / off available options in the more detailed selection process. The logic also implies that availability of data layers change; the user may only overlay data that was used to build the selected hazard model.

The user is presented with a pre-computed hazard map in the extent of the region showing the exceedance probability for a default IMT. The map extents over most 2/3 of the screen size including **map controls** and an **overview map** that can be toggled on / off. A **layer control** allows to overlay data sets specific for the PSHA model. To the left of the map display, M**odel Selection** in form of drop-down menus enable to choose various different parameter values. Below these, various **Export Utilities** are provided.

Map controls

The user can choose via the **map controls** his region of interest. The map controls use the MapServer-Technology and allow to:

- zoom in and out
- grab the map and move
- move using arrows
- fall back to global map extent
- turn on/off overview map

The **overview map** shows the extent of the map on the pre-computed map extent. The map can be toggled on and off.

Layer control:

In addition, different layers that display data used to compute the particular PSHA can be added. Examples of these data sets are:

- earthquake catalog
- Seismic source zonation
- earthquake activity rates
- b-values
- completeness levels

- prevailing type of faulting
- ground motion attenuation
- Maximum possible magnitude
- Maximum observed magnitude
- individual active faults
- composite seismogenic sources
- a regional overview of models implemented ground motion attenuation

Model selection

The hazard model to be displayed can be chose through various controls by drop-down menus. The choices are dependent on each other:

1. **PSHA model**:
Select the PSHA model version. For example, choose between different iteration of the PSHA model for the European-Mediterranean region that were produced during the FP-7 SHARE project. Additionally, national PSHA models or pre-existing models such as the SESAME-model could be displayed. *Default: Latest calculation*

2. **Investigation interval**:
Select between different time periods, such as 50 years. *Default: 50 years*

3. **Probability of Exceedance**:
Select probability of exceedance for the investigation interval. *Default: 10%*

4. **Intensity measure type (IMT)**:
Select IMT such as PGA [g], PGV [m/s], PGD [m], SA [g] at a particular frequency, EMS98 intensity and so on. *Default: PGA [g]*

5. **Site condition**:
Select between different soil condition if available. *Default: reference bedrock*

For a detailed specification on the access to single logic tree branches, details on the logic-tree definition are needed. This will be defined in future.

Export Utilities:

The export utilities give the user the opportunity to save what has been selected. The window shall be able to be minimized and maximized. Two options shall be implemented:

1. **KML export**:
Export the selected model and data as shown in the map to KML that can be displayed on Google Earth. The KML data shall also enable to choose data to be displayed or not on Google Earth, so it shall contain a tree utility to select.

2. **PDF export:**
Generate a one A4 page summary pdf-file. The pdf-file shall contain the map displayed on the web-browser without the control buttons on it. The pdf shall also contain information on what is displayed, so the choices from the model selection and the layer selection shall be printed nicely on the pdf.

3. Save data:
Save the pure data that is displayed in a common format (maybe just ascii) including a header with the information from the LOG-Window.

LOG-Window:

The LOG-Window displays the model and layer selections by the user. The window shall be a dynamic, so similarly to the **Overview map** the user can toggle it to a size that the content can be read or to only view it only as a button.

A.2 Usecase: Hazard Curve Portlet (HCP)

The users area of interest is to obtain information about the seismic hazard at a particular site.The portlet is designed in sections: In the left section, the user selects from multiple menus or provides input; the middle section shows an overview map such as shown in the HMP and orients about the site (indicated by some symbol); the right section shows the hazard curve or multiple curves. At the bottom of the page in the mid-section, **export utilities** shall be provided. In the right section, selected parameters are recorded and displayed in the **LOG-window**.

Site selection / Site condition:

The site can either be selected from the map window that provides the usual options as defined for the HMP. Selecting a site on the map with the mouse will be displayed in the coordinate boxes. Additionally, the user can choose between different soil conditions if available (*Default: reference bedrock)*

The button “Accept values” freezes the selection and invokes the logic behind to lookup further available data for this site.

Model selection:

This section provides multiple choices that depend on the pre-computed hazard results:

1. **PSHA Model**:
Select between different PSHA models if available for the site (see HMP)

2. **Type of hazard curve**:
Select between *Mean hazard curve*, *Mean hazard curve and fractiles* (one and two standard deviations if available), and *logic tree branch*. The latter is most complicated and implies more additional choices than the others. Default: *Mean hazard curve and fractiles (one standard deviation)*

3. **Investigation Interval**:
Select between different time periods (25y to 5000y). *Default: 50 years*

The next three choices are only needed for the choice *Logic tree branch in* **Type of hazard curve:**

4. **EQ Rupture Forecast**

5. **Source model**

6. **GMPE**

For a detailed specification on the access to single logic tree branches , details on the logic-tree definition are needed. This will be defined in future.

Curve Parameter:

The user selects from different available IMT and also selects the scale:

1. **IMT:**
Select IMT such as PGA [g], PGV [m/s], PGD [m], SA [g] at a particular frequency, EMS98 intensity and so on. *Default: PGA [g]*

2. **Scale:**
The y-axis shows always a probability of exceedance between 0 and 1. The x-axis varies with the selected IMT. Select linear, logarithmic on both axes, semi-logarithmic (x-logarithmic).

Export Utilities:

The export utilities give the user the opportunity to save what has been selected. The utility shall be able to minimize and maximize. Two options shall be implemented:

1. **KML export**:
Export the selected model and data as shown in the map to KML that can be displayed on Google Earth. The KML data shall also enable to choose data to be displayed or not on Google Earth, so it shall contain a tree utility to select.

2. **PDF export:**
Generate a one A4 page summary pdf-file. The pdf-file shall contain the map displayed on the web-browser without the control buttons on it. The pdf shall also contain information on what is displayed, so the choices from the model selection and the layer selection shall be printed nicely on the pdf.

3. **Save data:**
Save the pure data that is displayed in a common format (maybe just ascii) including a header with the information from the LOG-Window.

LOG-Window:

The LOG-Window displays the selections of the user. The window shall be dynamic, so similarly to the **Overview map** the user can toggle it to a size that the content can be read or to only view it only as a button.

A.3 Usecase: Hazard Spectra Portlet (HSP)

The user's area of interest is to obtain information about the seismic hazard at a particular site Figure 6. The portlet is designed in sections: In the left section, the user selects from multiple menus or provides input; the middle section shows an overview map such as shown in the HMP and orients about the site (indicated by some symbol); the right section shows the hazard spectra or multiple hazard spectra. At the bottom of the page in the mid-section, **export utilities** shall be provided. In the right section, selected parameters are recorded and displayed in the **LOG-window**.

Site selection / Site condition:

The site can either be selected from the map window that provides the usual options as defined for the HMP. Selecting a site on the map with the mouse will be displayed in the coordinate boxes. Additionally, the user can choose between different soil conditions if available (*Default: reference bedrock)*

The button “Accept values” freezes the selection and invokes the logic behind to lookup further available data for this site.

Model selection:

This section provides multiple choices that depend on the pre-computed hazard results:

1. **PSHA Model**:
Select between different PSHA models if available for the site (see HMP)

2. **Type of hazard curve**:
Select between *Mean hazard curve*, *Mean hazard curve and fractiles* (one and two standard deviations if available), and *logic tree branch*. The latter is most complicated and implies more additional choices than the others. Default: *Mean hazard curve and fractiles(one standard deviation)*

3. **Investigation Interval**:
Select between different time periods, such as 50 years. *Default: 50 years*

The next three choices are only needed for the choice *Logic tree branch in* **Type of hazard curve:**

4. EQ Rupture Forecast:

5. Source model:

6. GMPE:

For a detailed specification on the access to single logic tree branches , details on the logic-tree definition are needed. This will be defined in future.

Curve Parameter:

The user selects from different available measures and also selects the scale (*Period [s]*):

1. IML@Prob **or Prob@IML:**

Select between intensity-measure level (IML) at a given Probability of exceedance (y-scale: *SA [g]*) or vice versa a probability level at a given intensity-measure level (y-scale: *Probability of exceedance*) . *Default: Prob@IML*

2. **Scale:**

The y-axis shows always a probability of exceedance between 0 and 1. The x-axis stays at *Period [s]*. Select linear, logarithmic on both axes, semi-logarithmic (x-logarithmic).

Export Utilities:

The export utilities give the user the opportunity to save what has been selected. The utility shall be able to minimize and maximize. Two options shall be implemented:

1. **KML export**:

Export the selected model and data as shown in the map to KML that can be displayed on Google Earth. The KML data shall also enable to choose data to be displayed or not on Google Earth, so it shall contain a tree utility to select.

2. **PDF export:**

Generate a one A4 page summary pdf-file. The pdf-file shall contain the map displayed on the web-browser without the control buttons on it. The pdf shall also contain information on what is displayed, so the choices from the model selection and the layer selection shall be printed nicely on the pdf.

3. **Save data:**

Save the pure data that is displayed in a common format (maybe just ascii) including a header with the information from the LOG-Window.

LOG-Window:

The LOG-Window displays the selections of the user. The window shall be dynamic, so similarly to the **Overview map** the user can toggle it to a size that the content can be read or to only view it only as a button.

APPENDIX B Hazard Database Model – Table Layouts

In the following, specifications for additional portlets are describe that extend the functionalities, but are not basic requirements. It is designed as a wish-list that is technically possible but yet not assumed to be implemented in full detail during the project.

B.1 CALCULATIONGROUP (Calculation Group)

Table Name: CALCULATIONGROUP (Calculation Group)

Table Description: Stores Calculation Group Information. A Calculation Group may have one or more Calculation Owners

Primary Key: cgid

Name	Data Type	Comment
<u>cgcode</u>	Character(10) NOT NULL	Calculation Group Id, i.e. ETHZ
cgname	Character varying(50)	Name
cgdesc	Character varying(100)	Description
Cgauthlevel	Character(1)	Authorization Level
Cgadddate	timestamp without time zone	Addition Date
Cgremarks	Character varying(255)	Remarks

B.2 CALCULATIONOWNER (Calculation Owner)

Table Name: CALCULATIONOWNER (Calculation Owner)

Table Description: Stores Calculation Owner Information. A Calculation Owner belongs to only one Calculation Group. A Calculation Owner may instantiate a Hazard Calculation.

Primary Key: coid

Foreign Key: References CALCULATIONGROUP(cgid)

Name	Data Type	Comment
<u>cocode</u>	Character(10) NOT NULL	Calculation Owner Id, i.e. GEMUser

coname	Character varying(50)	Name
codesc	Character varying(100)	Description
coauthlevel	Character(1)	Authorization Level
coadddate	timestamp without time zone	Addition Date
coremarks	Character varying(255)	Remarks
cgcode	Character(10) NOT NULL	Foreign key – Calculation Group Id (CALCULATIONGROUP)

B.3 EARTHQUAKECATALOG (Earthquake Catalog)

Table Name: EARTHQUAKECATALOG (Earthquake Catalog)

Table Description: Stores Earthquake Catalogue Information. An Earthquake Catalog may have 0 or 2 EQCATCOMPLETENESS recorB. An earthquake catalogue has 1 or more events (EVENT) associated with it.

Primary Key: ecid (System-generated)

Name	Data Type	Comment
ecid	serial NOT NULL	Earthquake Catalog Id
ecprivatetag	Boolean	Private Data tag; Default to false; Values: true=for openGem authorized users only; false otherwise
ecshortname	Character(15)	Short Name
ecname	Character varying(50)	Name
ecdesc	Character varying(100)	Description
ecformattype	Integer	Format Type, Values: 1-ASCII,2-ESRI Shapefile, etc.
eccattype	Character(5)	Catalogue Type; Combinations of Values: H-Historical, I-Instrumental, S-Synthetic
ecstartdate	timestamp without time zone	Catalog Start Date
ecenddate	timestamp without time zone	Catalog End Date
ectimezone	Character(10)	Timezone; Values: GMT, UTC
eccvrgcd	Character(10)	Coverage Code; if country, country code, if region, region code
ecorigformatid	Integer	Original format id, to refer to another format specification table later
ecremarks	Character varying(255)	Remarks
ecpgareapolygon	geometry	Earthquake Catalogue area polygon in Postgis

		format
ecareapolygon	Character varying(5120)	Polygon in WKT String format
ecpgareamultipolygon	Geometry	Earthquake Catalogue area multipolygon in Postgis format
ecareamultipolygon	Character varying(5120)	Multipolygon in WKT String format

B.4 ECONSTANT (Equation Constant)

Table Name: ECONSTANT (Equation Constant)

Table Description: Stores Equation Constant Information (as may be defined in GMPE).

Primary Key: eccode

Name	Data Type	Comment
eccode	Character(5) NOT NULL	Equation Constant Code, i.e. R
ectypeid	integer	Equation Constant Data Type Id; Values 1-Integer, 2-Double, ...
ecshortname	Character(15)	Short Name, i.e. Distance
ecname	Character varying(50)	Name
ecdesc	Character varying(100)	Description
ecremarks	Character varying(255)	Remarks

B.5 ECREFERENCE (Earthquake Catalog Reference)

Table Name: ECREFERENCE (Earthquake Catalog Reference)

Table Description: Stores Earthquake Catalog Reference Information.

Primary Key: REFERENCELITERATURE(rlid) + EARTHQUAKECATALOG (ecid)

Foreign Key: References REFERENCELITERATURE(rlid)

References EARTHQUAKECATALOG (ecid)

Name	Data Type	Comment
rlid	integer NOT NULL	Reference Literature Id
ecid	integer NOT NULL	Earthquake Catalog Id
eradddt	timestamp without time zone	Addition Date

B.6 EQCATCOMPLETENESS (Earthquake Catalog Completeness)

Table Name: EQCATCOMPLETENESS (Earthquake Catalog Completeness)

Table Description: Stores Completeness Levels of Earthquake Catalogue Information. An Earthquake Catalog may have 0 or 2 EQCATCOMPLETENESS record.

Primary Key: ccid (System-generated) Foreign Key: References EARTHQUAKECATALOG(ecid)		
Name	Data Type	Comment
ccid	serial NOT NULL	Earthquake Cat Completeness Id
cctype	Character(1)	Completeness Type; Value: T-Time, A-Area
ccstartdate	timestamp without time zone	Start Date
ccenddate	timestamp without time zone	End Date
ccmlevel	Real	Catalog Completeness Magnitude Level
ecid	integer NOT NULL	Foreign key – Earthquake Catalog Id (EARTHQUAKECATALOG)
ccpgareapolygon	Geometry	Completeness Polygon in Postgis format
ccareapolygon	Character varying(1024)	Polygon in WKT String format
ccpgareamultipolygon	Geometry	Completeness MultiPolygon in Postgis format
ccareamultipolygon	Character varying(1024)	MultiPolygon in WKT String format

B.7 EVARIABLE (Equation Variable)

Table Name: EVARIABLE (Equation Variable) Table Description: Stores Equation Variable Information. Primary Key: evcode		
Name	Data Type	Comment
evcode	Character(5) NOT NULL	Equation Variable Code, i.e. A
evtypeid	integer	Equation Variable Data Type Id; Values 1-Integer, 2-Double, ...
evshortname	Character(15)	Short Name
evname	Character varying(50)	Name
evdesc	Character varying(100)	Description
evremarks	Character varying(255)	Remarks

B.8 EVENT (Event)

Table Name: EVENT (Event) Table Description: Stores Event Information. An Event belongs to exactly one Earthquake Catalog. An Earthquake Catalog may have 1 or more Events. Primary Key: evid (System-generated)

Foreign Key: References EARTHQUAKECATALOG(ecid)		
Name	Data Type	Comment
evid	serial NOT NULL	Event id
evshortname	Character(15)	Short name
evname	Character varying(50)	Name
evdesc	Character varying(100)	Description
evorigid	Character(50)	Id in original Earthquake Catalogue
evtimestamp	timestamp without time zone	Timestamp
evyear	Integer	Year of event
evmonth	Integer	Month of event
evday	Integer	Day of event
evhour	Integer	Hour of event
evmin	Integer	Minute of event
evsec	Integer	Second of event
evnanosec	Integer	Nanosecond of event
evlat	double precision	Latitude
evlong	double precision	Longitude
evdepth	double precision	Depth of event
evmagnitude	double precision	Magnitude of event (Mw)
evremarks	Character varying(255)	Remarks
evothdata1	Character varying(100)	Other Event data 1
evothdata2	Character varying(100)	Other Event data 2
evref	Character varying(100)	Event reference
everrorcode	integer DEFAULT 0	Error code
ecid	integer NOT NULL	Foreign key – Earthquake catalog id (EARTHQUAKECATALOG)
evpgpoint	Geometry	Event Point in Postgis format
evpoint	Character varying(255)	Point in WKT String format

B.9 GEFEATUREVALUE (GMPE Feature Value)

Table Name: GEFEATUREVALUE (GMPE Feature Value)

Table Description: Stores GMPE Feature Value Information.

Primary Key: GMPE(gecode) + GMPEFEATURE(gfcode)

Foreign Key: References GMPE(gecode)

References GMPEFEATURE(gfcode)

Name	Data Type	Comment
gecode	Character(10) NOT NULL	GMPE Code (GMPE)
gfcode	Character(10) NOT NULL	GMPE Feature Code (GMPEFEATURE)
gefvalstring	Character(50)	Value string
gefremarks	Character varying(255)	Remarks

B.10 GEOPOINT (Geographic Point)

Table Name: GEOPOINT (Geographic Point)

Table Description: Stores Geographic Point Information.

Primary Key: gpid (System-generated)

Foreign Key: References SITEAMPLIFICATION(sacode)

References SOILCLASS(socode)

Name	Data Type	Comment
gpid	serial NOT NULL	Geographic Point Id
gpname	Character varying(50)	Name
gpdesc	Character varying(100)	Description
sacode	Character(10)	Foreign key- Site Amplification Code (SITEAMPLIFICATION)
socode	Character(10)	Foreign key – Soil Classification Code (SOILCLASS)
gppgpoint	Geometry	Geographic Point in Postgis format
gppoint	Character(255)	Point in WKT String format

B.11 GEREFERENCE (GMPE Reference)

Table Name: GEREFERENCE (GMPE Reference)

Table Description: Stores GMPE Reference Literature Information.

Primary Key: REFERENCELITERATURE(rlid) + GMPE(geid)

Foreign Key: References REFERENCELITERATURE(rlid)

References GMPE(geid)

Name	Data Type	Comment
rlid	integer NOT NULL	Reference Literature Id (REFERENCELITERATURE)
geid	integer NOT NULL	GMPE Id (GMPE)
gradddt	timestamp without time zone	Addition Date

B.12 GMECONSTANT (GMPE Constant)

Table Name: GMECONSTANT (GMPE Constant)
Table Description: Stores GMPE Constant Information.
Primary Key: GMPE(geid) + ECONSTANT(eccode)
Foreign Key: References GMPE(geid)
References ECONSTANT(eccode)

Name	Data Type	Comment
geid	integer NOT NULL	GMPE Id (GMPE)
eccode	Character(10) NOT NULL	Equation Constant Code (ECONSTANT)
gmecvalue	Character (20)	Equation Constant Value
gmecremarks	Character varying(255)	Remarks

B.13 GMEVARIABLE (GMPE Variable)

Table Name: GMEVARIABLE (GMPE Variable)
Table Description: Stores GMPE Variable Information.
Primary Key: GMPE(geid) + EVARIABLE(evcode)
Foreign Key: References GMPE(geid)
References EVARIABLE(evcode)

Name	Data Type	Comment
geid	integer NOT NULL	GMPE Id (GMPE)
evcode	Character(5) NOT NULL	Equation Variable Code (EVARIABLE)
gmevremarks	Character varying(255)	Remarks

B.14 GMPARAMVALUE (GMPE Parameter Value)

Table Name: GMPARAMVALUE (GMPE Parameter Value)
Table Description: Stores GMPE Parameter Value Information.
Primary Key: GMPE(gecode) + GMPEPARAMETER(gpcode)

Foreign Key: References GMPE(gecode) References GMPEPARAMETER(gpcode)		
Name	Data Type	Comment
gecode	Character(10) NOT NULL	GMPE Code (GMPE)
gpcode	Character(10) NOT NULL	GMPE Parameter Code (GMPEPARAMETER)
gepvalstring	Character varying(50)	Value string
gepremarks	Character varying(255)	Remarks

B.15 GMPE (Ground Motion Prediction Equation)

Table Name: GMPE (Ground Motion Prediction Equation) Table Description: Stores GMPE Information (also called Attenuation Relations). Primary Key: gecode Foreign Key: References SEISMOTECENVT(secode)		
Name	Data Type	Comment
gecode	Character(10) NOT NULL	GMPE Code
geprivatetag	Boolean	GMPE Private tag: true=for GEM use only, false=otherwise
geshortname	Character(15)	Short Name
gename	Character varying(50)	Name
gedesc	Character varying(100)	Description
geequation	Character varying(5120)	GMPE Equation in Functional form
geeqndbdefntag	Boolean	GMPE Equation Definition Tag: true=defined in db (in standard format), false=not defined
geeqntypecode	Character(1)	Ground Motion Prediction Equation Type code; i.e. Standard=3 constants, 3 vars; other=see vars, const
geremarks	Character varying(255)	Remarks
gtcode	Character(15)	Foreign Key - GMPE Type Code (GMPE)
gepgareapolygon	Geometry	GMPE Area Polygon or Region of Validity of GMPE, in Postsgis format
geareapolygon	Character varying(5120)	Polygon in WKT String format
gepgareamultipolygon	Geometry	GMPE Area Multipolygon or Region of Validity of GMPE, in Postsgis format

geareamultipolygon	Character varying(5120)	MultPolygon in WKT String format
secode	Character(10)	Foreign key – Seismotectonic Environment Code (SEISMOTECENVT)

B.16 GMPEFEATURE (GMPE Feature)

Table Name: GMPEFEATURE (GMPE Feature)
Table Description: Stores GMPE Feature Information.
Primary Key: gfcode

Name	Data Type	Comment
gfcode	Character(10) NOT NULL	GMPE Feature Code
gftypeid	integer	GMPE Feature Data Type Id; Values 1-Integer, 2-Double, ...
gfpossvalstring	Character varying (2048)	Possible Value String
gfshortname	Character(20)	Short Name
gfname	Character varying(50)	Name
gfdesc	Character varying(100)	Description
gfremarks	Character varying(255)	Remarks

B.17 GMPEPARAMETER (GMPE Parameter)

Table Name: GMPEPARAMETER (GMPE Parameter)
Table Description: Stores GMPE Parameter Information.
Primary Key: gpcode

Name	Data Type	Comment
gpcode	Character(10) NOT NULL	GMPE Parameter Code
gptypeid	integer	GMPE Parameter Data Type Id; Values 1-Integer, 2-Double, ...
gppossvalstring	Character varying (2048)	Possible Value String
gpshortname	Character(20)	Short Name
gpname	Character varying(50)	Name
gpdesc	Character varying(100)	Description
gpremarks	Character varying(255)	Remarks

B.18 HAZARDCALCULATION (Hazard Calculation)

Table Name: HAZARDCALCULATION (Hazard Calculation)

Table Description: Stores Hazard Calculation Information.

Primary Key: hcid (System-generated)

Foreign Key: References CALCULATIONOWNER(cocode)

References HAZARDSOFTWARE(hscode)

References HAZARDINPUTBASICMODEL(hibmid)

References HAZARDINPUTLTREEMODEL(hilmid)

References HILMPATH(hilmpid)

References EVENT(evid)

Name	Data Type	Comment
hcid	serial NOT NULL	Hazard Calculation id
hcshortname	character(20)	Short Name
hcname	character varying(50)	Name
hcdesc	character varying(100)	Description
hcstarttimestamp	timestamp without time zone	Start Timestamp of Calculation
hcendtimestamp	timestamp without time zone	End Timestamp of Calculation
hcprobdettag	character(1)	Tag for Probabilistic (P) or Deterministic (C) Calculation
hcgemgentag	Boolean	GEM generated tag; true if GEM-generated, false otherwise
hcremarks	character varying(255)	Remarks
hscode	character(10)	Foreign key – Hazard Software Code (HAZARDSOFTWARE)
cocode	character(10)	Foreign key – Calculation Owner Code (CALCULATIONOWNER)
hibmid	Integer	Foreign key – Hazard Input Basic Model Id (HAZARDINPUTBASICMODEL)
hilmid	Integer	Foreign key – Hazard Input Logic Tree Model Id (HAZARDINPUTLOGICTREEMODEL)
hilmpid	Integer	Foreign key – Hazard Input Logic Tree Model Path Id (HILMPATH)
evid	Integer	Foreign key – Event Id (EVENT)
hcpgareapolygon	Geometry	Hazard Calculation Area Polygon in Postgis Format

hcareapolygon	character varying(5120)	Polygon in WKT String format
hcpgareamultipolygon	geometry	Hazard Calculation Area Multipolygon in Postgis Format
hcareamultipolygon	character varying(5120)	Multipolygon in WKT String format

B.19 HAZARDCURVE (Hazard Curve)

Table Name: HAZARDCURVE (Hazard Curve)

Table Description: Stores Hazard Curve Information. A hazard curve shows the probability of exceeding different ground motion values at a location. For example, the 2% probability of exceedance in 50 years is one point on a hazard curve.

Primary Key: hcrvid (System-generated)

Foreign Key: References GEOPOINT(gpid)

References HAZARDCALCULATION(hcid)

References INTENSITYMEASURETYPE(imcode)

Name	Data Type	Comment
hcrvid	serial NOT NULL	Hazard Curve Id
hcrvshortname	character(20)	Short Name
hcrvname	character varying(50)	Name
hcrvdesc	character varying(100)	Description
hcrvremarks	character varying(255)	Remarks
hcrvtimestamp	timestamp without time zone	Timestamp of Hazard Curve generation
hcrvmingrdmotion	Double precision	Hazard Curve Minimum Ground Motion
hcrvmaxgrdmotion	Double precision	Hazard Curve Maximum Ground Motion
hcrvtimeperiod	Integer	Time period for a Hazard Curve
hcrvsadamping	Integer	Spectral acceleration damping value
hcrvsaperiod	Integer	Spectral acceleration period
gpid	Integer	Foreign key – Geographic Point Id (GEOPOINT)
hcid	Integer	Foreign key – Hazard Calculation Id (HAZARDCALCULATION)
imcode	character (10)	Foreign key – Intensity Measure Type Code (INTENSITYMEASURETYPE)

B.20 HAZARDINPUTBASICMODEL (Hazard Input Basic Model)

Table Name: HAZARDINPUTBASICMODEL (Hazard Input Basic Model)

Table Description: Stores Hazard Input Basic Model Information.

Primary Key: hibmid (System-generated)

Foreign Key: References SOURCEGEOMETRYCATALOG(scid)

Name	Data Type	Comment
hibmid	serial NOT NULL	Hazard Input Basic Model Id
hibmshortname	character(20)	Short Name
hibmname	character varying(50)	Name
hibmdesc	character varying(100)	Description
hibmremarks	character varying(255)	Remarks
hibmcvrgtypecode	character(1)	Coverage Type; 1- all, 2- area, 3-chosen sources
scid	Integer	Foreign key – Source Geometry Catalog Id (SOURCEGEOMETRYCATALOG)
hibmpgareapolygon	geometry (polygon)	Hazard Input Basic Model Area Polygon in Postgis format
hibmareapolygon	character varying(5120)	Polygon in WKT String format
hibmpgareamultipolygon	geometry (multipolygon)	Hazard Input Basic Model Area Multipolygon in Postgis format
hibmareamultipolygon	character varying(5120)	Multipolygon in WKT String format

B.21 HAZARDINPUTLTREEMODEL (Hazard Input Logic Tree Model)

Table Name: HAZARDINPUTLTREEMODEL (Hazard Input Logic Tree Model)

Table Description: Stores Hazard Input Logic Tree Model Information.

Primary Key: hilmid (System-generated)

Foreign Key: References LOGICTREESTRUC(ltsid)

Name	Data Type	Comment
hilmid	serial NOT NULL	Hazard Input Logic Tree Model id
hilmshortname	character(15)	Short Name
hilmname	character varying(50)	Name
hilmdesc	character varying(100)	Description
hilmremarks	character varying(255)	Remarks
hilmcvrgtype	character(1)	Coverage Type; 1- all, 2- area, 3-chosen sources

ltsid	Integer NOT NULL	Foreign key – Logic Tree Structure Id (LOGICTREESTRUC)
hilmpgareapolygon	geometry (polygon)	Hazard Input Logic Tree Model Area Polygon in Postgis format
hilmareapolygon	character varying(5120)	Polygon in WKT String format
hilmpgareamultipolygon	geometry (multipolygon)	Hazard Input Logic Tree Model Area Multipolygon in Postgis format
hilmareamultipolygon	character varying(5120)	Multipolygon in WKT String format

B.22 HAZARDMAP (Hazard Map)

Table Name: HAZARDMAP (Hazard Map)

Table Description: Stores Hazard Map Information. For example, a hazard map can depict probabilistic values of peak horizontal ground acceleration (PGA) and spectral accelerations (SA) at 0.2, 0.3, 1.0 second periods (5% of critical damping) with 10%, 5% and 2% probabilities of exceedance in 50 years.

Primary Key: hmapid (System-generated)

Foreign Key: References HAZARDCALCULATION(hcid)

References INTENSITYMEASURETYPE(imcode)

Name	Data Type	Comment
hmapid	serial NOT NULL	Hazard Map Id
hmapshortname	character(15)	Short Name
hmapname	character varying(50)	Name
hmapdesc	character varying(100)	Description
hmapremarks	character varying(255)	Remarks
hmaptimestamp	timestamp without time zone	Timestamp of Hazard Map generation
hmaptimedepstartdate	timestamp without time zone	Start Date for Time-dependent Hazard Map
hmaptimedependdate	timestamp without time zone	End Date for Time-dependent Hazard Map
hmapexceedprobpct	Double precision	Exceedance Probability Percent
hmapexceedyears	Integer	Exceedance Number of Years
hmreturnperiod	Integer	Return Period
hmapdamping	Integer	Damping
hmapgridsize	Float	Grid size for map
hcid	Integer	Foreign key – Hazard Calculation Id (HAZARDCALCULATION)

imcode	character(10)	Foreign key – Intensity Measure Type Code (INTENSITYMEASURETYPE)

B.23 HAZARDPOINTVALUE (Hazard Point Value)

Table Name: HAZARDPOINTVALUE (Hazard Point Value)

Table Description: Stores Hazard Point Value Information.

Primary Key: hpid (System-generated)

Foreign Key: References HAZARDCALCULATION(hcid)

References GEOPOINT(gpid)

Name	Data Type	Comment
hpid	serial NOT NULL	System-generated sequence id
hpvalue	double precision	Intensity Measure Value
hpexceedprobpct	double precision	Exceedance Probability Percent
hpexceedyears	Integer	Exceedance Number of Years
hcid	Integer	Foreign key – Hazard Calculation Id (HAZARDCALCULATION)
gpid	Integer	Foreign key – Geographic Point Id (GEOPOINT)
imcode	Character(10)	Foreign key – Intensity Measure Type Code (INTENSITYMEASURETYPE)

B.24 HAZARDSOFTWARE (Hazard Software)

Table Name: HAZARDSOFTWARE (Hazard Software)

Table Description: Stores Hazard Software Information.

Primary Key: hsid

Name	Data Type	Comment
hscode	Character(10) NOT NULL	Hazard Software Id, i.e. OPENSHA
hsname	Character varying(50)	Name
hsdesc	Character varying(100)	Description
hsadddate	timestamp without time zone	Addition Date
hsremarks	Character varying(255)	Remarks

B.25 HIBMGE (Hazard Input Basic Model GMPE)

Table Name: HILMGE (Hazard Input Basic Model GMPE)

Table Description: Stores Hazard Input Basic Model GMPE Information.

Primary Key: HAZARDINPUTBASICMODEL(hibmid) + GMPE(gecode) Foreign Key: References HAZARDINPUTBASICMODEL(hibmid) References GMPE(gecode)		
Name	Data Type	Comment
hibmid	integer NOT NULL	Hazard Input Basic Model Id (HAZARDINPUTBASICMODEL)
gecode	integer NOT NULL	GMPE Code (GMPE)
hibmgeweight	double precision	Weight of GMPE

B.26 HIBMREFERENCE (Hazard Input Basic Model Reference)

Table Name: HIBMREFERENCE (Hazard Input Basic Model Reference) Table Description: Stores Hazard Input Basic Model Reference Information. Primary Key: REFERENCELITERATURE(rlid) + HAZARDINPUTBASICMODEL(hibmid) Foreign Key: References REFERENCELITERATURE(rlid) References HAZARDINPUTBASICMODEL(hibmid)		
Name	Data Type	Comment
rlid	integer NOT NULL	Reference Literature Id (REFERENCELITERATURE)
hibmid	integer NOT NULL	Hazard Input Basic Model Id (HAZARDINPUTBASICMODEL)
hibmradddt	timestamp without time zone	Addition Date

B.27 HILMPATH (Hazard Input Logic Tree Model Path)

Table Name: HILMPATH (Hazard Input Logic Tree Model Path) Table Description: Stores Hazard Input Logic Tree Model Path Information. One logic tree path uses one source geometry catalog (source model). Primary Key: hilmpid (System-generated) Foreign Key: References HAZARDINPUTLOGICTREEMODEL(hilmid) References SOURCEGEOMETRYCATALOG(scid)		
Name	Data Type	Comment
hilmpid	serial NOT NULL	Hazard Input Logic Tree Model Path Id
hilmpshortname	Character(15)	Short Name
hilmpname	Character varying(50)	Name
hilmpdesc	Character varying(100)	Description

hilmpremarks	Character varying(255)	Remarks
hilmppathstring	character(1024)	Path string, in format: <ParamType>_<ParamValue>_<BranchWeight>, i.e. FaultModel_GR_18.DipUncertainty_ 40degrees_20.RecurUncertaintyModel_12mmPerYear_100
hilmltree	Ltree	Logic Tree path using Ltree (Postgresql contrib) structure
hilmpfinalpathtag	Boolean	True if final path, False otherwise
hilmpweight	double precision	Final weight for logic tree path
hilmid	integer NOT NULL	Foreign key – Hazard Input Logic Tree Model Id (HAZARDINPUTLOGICTREEMODEL)
scid	Integer	Foreign key – Source Geometry Catalog Id (SOURCEGEOMETRYCATALOG)

B.28 HILMPGE (Hazard Input Logic Tree Model Path GMPE)

Table Name: HILMPGE (Hazard Input Logic Tree Model Path GMPE)

Table Description: Stores Hazard Input Logic Tree Model Path GMPE Information.

Primary Key: HILMPATH(hilmpid) + GMPE(gecode)

Foreign Key: References HILMPATH(hilmpid)
References GMPE(gecode)

Name	Data Type	Comment
hilmpid	integer NOT NULL	Hazard Input Logic Tree Model Id (HAZARDINPUTLTREEMODEL)
gecode	Character(10) NOT NULL	GMPE Code (GMPE)
hilmpgeweight	double precision	Weight of GMPE

B.29 HILMREFERENCE (Hazard Input Logic Tree Model Reference)

Table Name: HILMREFERENCE (Hazard Input Logic Tree Model Reference)

Table Description: Stores Hazard Input Logic Tree Model Reference Information.

Primary Key: REFERENCELITERATURE(rlid) + HAZARDINPUTLTREEMODEL(hilmid)

Foreign Key: References REFERENCELITERATURE(rlid)
References HAZARDINPUTLTREEMODEL(hilmid)

Name	Data Type	Comment
rlid	integer NOT NULL	Reference Literature Id (REFERENCELITERATURE)
hilmid	integer NOT NULL	Hazard Input Logic Tree Model Id

		(HAZARDINPUTLTREEMODEL)
hilmradddt	timestamp without time zone	Addition Date

B.30 HILMRULESET (Hazard Input Logic Tree Model Ruleset)

Table Name: HILMRULESET (Hazard Input Logic Tree Model Ruleset)

Table Description: Stores Hazard Input Logic Tree Model Ruleset Information.

Primary Key: hilmrid (System-generated)

Foreign Key: References HAZARDINPUTLTREEMODEL(hilmid)
References LTREEPARAMTYPE(ltptid)
References LTREEPARAMVALUE(ltpvid)

Name	Data Type	Comment
hilmrid	serial NOT NULL	Hazard Input Logic Tree Model Ruleset Id
hilmrcvrgtypecode	Integer	Coverage Type; Values: 1-All sources, 2-Area, 3-Some sources
hilmractiontypecode	Integer	Action Type; Values: 1-Replace, 2-Add, 3-do nothing
hilmrvalstring	Character(100)	Ruleset Value string
ltptid	integer NOT NULL	Foreign key – Logic Tree Parameter Type Id (LTREEPARAMTYPE)
ltpvid	integer NOT NULL	Foreign key – Logic Tree Parameter Value Id (LTREEPARAMVALUE)
hilmid	integer NOT NULL	Foreign key – Hazard Input Logic Tree Model Id (HAZARDINPUTLTREEMODEL)

B.31 INTENSITYMEASURETYPE (Intensity Measure Type)

Table Name: INTENSITYMEASURETYPE (Intensity Measure Type)

Table Description: Stores Intensity Measure Type Information such as Peak Ground Acceleration (PGA), Peak Ground Velocity (PGV), Peak Ground Displacement (PGD), Modified Mercalli Intensity (MMI), Spectral Acceleration (i.e. SA 0.2, SA 1.0).

Primary Key: imcode

Name	Data Type	Comment
imcode	Character(10) NOT NULL	Intensity Measure Type Id, i.e. PGA, PGV, MMI
imname	Character varying (50)	Name, i.e. Peak Ground Acceleration
imdesc	Character varying (100)	Description
imvaluemin	double precision	Minimum Acceptable Value (for range-checking), i.e. 0
imvaluemax	double precision	Maximum Acceptable Value (for range-checking), i.e. 10

imunittype	Character(10)	Unit type, i.e. g (gravity), cms/sec (cgs)
imunitdescr	Character varying(100)	Unit description
imremarks	Character varying(255)	Remarks

B.32 LOGICTREESTRUC (Logic Tree Structure)

Table Name: LOGICTREESTRUC (Logic Tree Structure)

Table Description: Stores Logic Tree Structure Information. The Logic Tree Structure gives the specification for particular logic tree structures.

Primary Key: ltsid (System-generated)

Name	Data Type	Comment
ltsid	serial NOT NULL	Logic Tree Structure Id
ltsname	Character varying (50)	Name
ltsdesc	Character varying (100)	Description
ltsremarks	Character varying(255)	Remarks

B.33 LTREEPARAMTYPE (Logic Tree Parameter Type)

Table Name: LTREEPARAMTYPE (Logic Tree Parameter Type)

Table Description: Stores Logic Tree Parameter Type Information. Parameter types are the characteristics defined on each level of logic trees. For example, in the USGS NSHM 2008 WUS Model, some parameter types defined are Fault Model, Dip Uncertainty Model, Magnitude Uncertainty Model, Ground Motion Model.

Primary Key: ltptid (System-generated)

Name	Data Type	Comment
ltptid	serial NOT NULL	Logic Tree Parameter Type Id
ltptshortname	Character(20)	Short Name, i.e. FaultMdl, DipUncertMdl
ltptname	Character varying (50)	Name, i.e. Fault Model, Dip Uncertainty Model
ltptdesc	Character varying (100)	Description
ltptremarks	Character varying(255)	Remarks
ltptdatatypecode	Integer	Data type, ex. 1-Integer, 2-Double; information needed for value replacement/computation
ltptpossvalstring	Character varying(2048)	Possible Value String
ltptvaluemin	double precision	Minimum value for range checking
ltptvaluemax	double precision	Maximum value for range checking
ltptmapping	Character varying(255)	Mapping to actual fields on file (for later implementation)

B.34 LTREEPARAMTYPELEVEL (Logic Tree Parameter Type Level)

Table Name: LTREEPARAMTYPELEVEL (Logic Tree Parameter Type Level)

Table Description: Stores Logic Tree Parameter Type Level Information. Root is level 0, 1st set of branching is level 1, and so on, until level n.

Primary Key: LOGICTREESTRUC(ltsid) + LTREEPARAMTYPE(ltptid)

Foreign Key: References LOGICTREESTRUC(ltsid)
References LTREEPARAMTYPE(ltptid)

Name	Data Type	Comment
ltsid	integer NOT NULL	Logic Tree Structure Id (LOGICTREESTRUC)
ltptid	integer NOT NULL	Logic Tree Parameter Type Id (LTREEPARAMTYPE)
ltptlevel	Integer	Level of Logic Tree Parameter Type
ltptnumbranches	Integer	Number of branches for given level
ltptbranchsettag	Boolean	True-default all parameter values set in level, False-otherwise

B.35 LTREEPARAMVALUE (Logic Tree Parameter Value)

Table Name: LTREEPARAMVALUE (Logic Tree Parameter Value)

Table Description: Stores Logic Tree Parameter Value Information. Parameter values are specified values of Parameter Types. For example, in the USGS NSHM 2008 WUS Model, the parameter values for the Fault Model parameter type are Characteristic and Gutenberg-Richter; the parameter values for the Dip Uncertainty Model parameter type are 40 degrees, 50 degrees and 60 degrees.

Primary Key: ltpvid (System-generated)

Foreign Key: References LTREEPARAMTYPE(ltptid)

Name	Data Type	Comment
ltpvid	serial NOT NULL	Logic Tree Parameter Value Id
ltpvshortname	Character (20)	Name, i.e. CHAR, GR (for Fault Model parameter type)
ltpvname	Character varying (50)	Name, i.e. Characteristic, Gutenberg-Richter (for Fault Model parameter type)
ltpvdesc	Character varying (100)	Description
ltpvremarks	Character varying(255)	Remarks
ltptid	integer NOT NULL	Foreign key – Logic Tree Parameter Type (LTREEPARAMTYPE)

B.36 MAGFREQDISTN (Magnitude Frequency Distribution)

Table Name: MAGFREQDISTN (Magnitude Frequency Distribution)

Table Description: Stores Magnitude Frequency Distribution Information. This is a function giving the rate of events (per year) on the Y axis versus magnitude on the X axis (Refer: www.opensha.org/documentation/glossary). Primary Key: mfdcode		
Name	Data Type	Comment
mfdcode	character(15) NOT NULL	Magnitude Frequency Distribution Code, i.e. GR for Gutenberg-Richter, CHAR for characteristic distribution/Youngs and Coppersmith distribution
mfdname	character varying (50)	Name
mfddesc	character varying (100)	Description
mfdremarks	character varying(255)	Remarks
mfddisctag	Boolean	Dicrete Distribution Tag; True-discrete distribution, False-otherwise

B.37 MAGRUPTURERELATION (Magnitude Rupture Relation)

Table Name: MAGRUPTURERELATION (Magnitude Rupture Relation) Table Description: Stores Magnitude Rupture Relation Information, also equivalent to Magnitude Scaling Relationship. Primary Key: mrrcode		
Name	Data Type	Comment
mrrcode	character(15) NOT NULL	Magnitude Rupture Relation Code, i.e. WECO1994 for Wells and Coppersmith 1994, HABA2002 for Hank and Bakun 2002
mrrname	character varying (50)	Name
mrrdesc	character varying (100)	Description
mrrremarks	character varying(255)	Remarks

B.38 REFERENCELITERATURE (Reference Literature)

Table Name: REFERENCELITERATURE (Reference Literature) Table Description: Stores Reference Literature Information. Primary Key: rlid (System-generated)		
Name	Data Type	Comment
rlid	serial NOT NULL	Reference Literature
rlshortname	character(15)	Short Name
rlmainauthorfname	character(25)	Author First Name
rlmainauthorlname	character(25)	Author Last Name

rlotherauthor	character varying(512)	Other Authors
rltitle	character varying(512)	Title
rlmediatype	character(1)	Media Type, i.e. J-Journal, B-Book, P-Periodical, W-Webpage
rlperiodicaltitle	character varying(512)	Periodical Title
rlpublishercity	character varying(100)	Publisher City
rlpublishername	character varying(512)	Publisher Name
rlpubnyear	Integer	Publication Year
rlvolnum	Integer	Volume
rlissuenum	Integer	Issue Number
rlpagenums	character varying(100)	Page numbers, i.e. 10-25
rllastaccessdate	Date	Last Access Date
rltype	character(2)	Reference Literature Type
rlurl	character varying(1024)	URL (Universal Resource Locator)
rlremarks	character varying(255)	Remarks

B.39 SCREFERENCE (Source Geometry Catalog Reference)

Table Name: SCREFERENCE (Source Geometry Catalog Reference)

Table Description: Stores Source Geometry Catalog Reference Information.

Primary Key: REFERENCELITERATURE(rlid) + SOURCEGEOMETRYCATALOG (scid)

Foreign Key: References REFERENCELITERATURE(rlid)

References SOURCEGEOMETRYCATALOG (scid)

Name	Data Type	Comment
rlid	integer NOT NULL	Reference Literature Id (REFERENCELITERATURE)
scid	integer NOT NULL	Source Geometry Catalog Id (SOURCEGEOMETRYCATALOG)
scradddt	timestamp without time zone	Addition Date

B.40 SEISMICSOURCE (Seismic Source)

Table Name: SEISMICSOURCE (Seismic Source)

Table Description: Stores Seismic Source Information. Currently, we define a seismic source to be either: (1) a fault, (2) an area source, (3) a gridded seismicity point or (4) a subduction fault. In terms of geometry, a fault is in the form of a multilinestring, an area source is in the form of a polygon or multipolygon, a gridded seismicity point is in the form of a

point, and a subduction fault consists of a top and bottom multilinestring. A seismic source belongs to exactly one Source Geometry Catalogue. A Source Geometry Catalogue may have 1 or more Seismic Sources. Currently designed as 2-dimensional sources – the system has support to handle the storage of 3-dimensional sources. Primary Key: ssid (System-generated) Foreign Key: References SOURCEGEOMETRYCATALOG(scid) References SEISMOTECENVT(secode)		
Name	Data Type	Comment
ssid	serial NOT NULL	Seismic Source Id
ssrctypecode	Integer	Seismic Source Type: 1-fault,2-area source, 3-gridded seismicity point; 4-subduction fault (Defined in org.opengem.db.hazard.enums.SeismicSourceTypeCode)
ssgeomtypecode	Integer	Seismic Source Geometry Type Code: 1-Point, 2-Multilinestring, 3-Polygon, 4-Multipolygon, 5-Point3D, 6-Multilinestring3D, 7-Polygon3D, 8-Multipolygon3D (3D geometries for later implementation: Defined in org.opengem.db.hazard.enums. SeismicSourceGeomeTypeCode)
ssgrdefaulttag	Boolean	Gutenberg-Richter (GR) distribution default tag: True-if default to GR, False-otherwise
ssorigid	character(50)	Seismic Source Id value in original Source Geom Catalogue
ssshortname	character(15)	Short Name
ssname	character varying(50)	Name
ssdesc	character varying(100)	Description
ssremarks	character varying(255)	Remarks
ssarea	double precision	Computed area of source (if area source)
ssanormalized	double precision	Computed normalized a value
ssdepth	double precision	Depth of seismic source
ssbackgrdzonetag	boolean	Background Zone Tag; True-background zone, False-otherwise
sserrorcode	Integer	Error Code
sspgpoint	geometry	Point in Postgis format (filled up if ssgeomtypecode is Point)

sspoint	character varying(255)	Point in WKT String format (filled up if ssgeomtypecode is Point)
sspgmultilinestring	geometry	Fault Multilinestring in Postgis format (filled up if ssgeomtypecode is Multilinestring)
ssmultilinestring	character varying(5120)	Multilinestring in WKT String format (filled up if ssgeomtypecode is Multilinestring)
sspgpolygon	geometry	Seismic Source Zone Polygon in Postgis format (filled up if ssgeomtypecode is Polygon)
sspolygon	character varying(5120)	Polygon in WKT String format (filled up if ssgeomtypecode is Polygon)
sspgmultipolygon	geometry	Seismic Source Zone Multipolygon in Postgis format (filled up if ssgeomtypecode is Multipolygon)
ssmultipolygon	character varying(5120)	Multipolygon in WKT String format (filled up if ssgeomtypecode is Multipolygon)
sspgtopmultilinestring	geometry	Subduction Fault Top Multilinestring in Postgis format (filled up if ssgeomtypecode is Multilinestring)
sstopmultilinestring	character varying(5120)	Subduction Fault Top Multilinestring in WKT String format (filled up if ssgeomtypecode is Multilinestring)
sspgbottommultilinestring	geometry	Subduction Fault Bottom Multilinestring in Postgis format (filled up if ssgeomtypecode is Multilinestring)
ssbottommultilinestring	character varying(5120)	Subduction Fault Bottom Multilinestring in WKT String format (filled up if ssgeomtypecode is Multilinestring)
scid	integer NOT NULL	Foreign key – Source Geometry Catalog Id (SOURCEGEOMETRYCATALOG)
secode	character(10)	Foreign key – Seismotectonic Environment Code (SEISMOTECENVT)

B.41 SEISMOTECENVT (Seismotectonic Environment)

Table Name: SEISMOTECENVT (Seismotectonic Environment)

Table Description: Stores Seismotectonic Environment Information.

Primary Key: secode

Name	Data Type	Comment
secode	Character(10) NOT NULL	Seismotectonic Environment Code, i.e. ACTIVE,

		SCR, SUBDUCTION, VRANCEATYP, VOLCANIC, UNCLASSIFIED
sename	Character varying(50)	Name, i.e. Active, Stable Continental Region, Subduction
sedesc	Character varying(100)	Description
seremarks	Character varying(255)	Remarks

B.42 SFAULTCHAR (Seismic Source Fault Characteristics)

Table Name: SFAULTCHAR (Seismic Source Fault Characteristics)

Table Description: Stores Seismic Source Fault Characteristics Information. Contains other information that is specific to a fault. Each SEISMICSOURCE (if fault) may have 0 or 1 SFAULTCHAR records.

Primary Key: ssfcid (System-generated)

Foreign Key: References SEISMICSOURCE(ssid)

Name	Data Type	Comment
ssfcid	serial NOT NULL	Seismic Source Fault Characteristic Id
ssfcstatus	character(2)	Fault status Code: A-Active, I-Inactive
ssfcsliprate	double precision	Slip Rate
ssfcslipratesd	double precision	Slip Rate Standard Deviation
ssfcfloattypecode	Integer	Float Type Code, i.e. 1-Stirling
ssfcfloatingruptureflag	Boolean	Floating Rupture Flag, True if floating rupture , False otherwise
ssfcfloatoffsetalongstrike	Integer	Float offset along Strike, in degrees
ssfcfloatoffsetalongdip	Integer	Float offset along Dip, in degrees
ssfcrupturetop	Integer	Rupture Top, in kilometers, i.e. 0
ssfcrupturebottom	Integer	Rupture Bottom, in kilometers, i.e. 15
ssid	integer NOT NULL	Foreign key – Seismic Source Id (SEISMICSOURCE)

B.43 SITEAMPLIFICATION (Site Amplification)

Table Name: SITEAMPLIFICATION (Site Amplification)

Table Description: Stores Site Amplification Information.

Primary Key: sacode

Name	Data Type	Comment

sacode	Character(10) NOT NULL	Site Amplification Code, i.e. SOFTROCK
saname	Character varying(50)	Name, i.e. Soft Rock
sadesc	Character varying(100)	Description
saremarks	Character varying(255)	Remarks
savs30min	double precision	Site Amplification VS 30 Minimum Value, i.e. 800 m/sec
savs30max	double precision	Site Amplification VS 30 Maximum Value, i.e. 3000 m/sec
savs30descstring	Character varying (50)	Site Amplification Description String
sanehrp	Character(4)	NEHRP Classification: Values A, AB, B, BC, C, CD, D, DE, E
saintampl	Real	Intensity Amplification

B.44 SOILCLASS (Soil Class)

Table Name: SOILCLASS (Soil Class)

Table Description: Stores Soil Class Information.

Primary Key: socode

Name	Data Type	Comment
socode	Character(2) NOT NULL	Soil Class Code f(follows SIA 261): Values A, B, C, D, E, F1, F2
soname	Character varying(50)	Name
sodesc	Character varying(100)	Description
soremarks	Character varying(255)	Remarks
sovalue	Character(10)	Value

B.45 SOURCEGEOMETRYCATALOG (Source Geometry Catalogue)

Table Name: SOURCEGEOMETRYCATALOG (Source Geometry Catalogue)

Table Description: Stores Source Geometry Catalogue Information. A Source Geometry Catalog has 1 or more seismic sources (SEISMICSOURCE) associated with it.

Primary Key: scid (System-generated)

Name	Data Type	Comment
scid	serial NOT NULL	Source Geometry Catalog Id
scprivatetag	Boolean	Private tag: True-for private/OpenGem use only, False- otherwise

scshortname	Character(15)	Short Name
scname	Character varying(50)	Name
scdesc	Character varying(100)	Description
scremarks	Character varying(255)	Remarks
sctypecode	Character(5)	Source Geometry Catalog Type Code: Either H-Historic, I-Instrumental, S-Synthetic or combinations of types
scstartdate	timestamp without time zone	Start Date of Catalog Information
scenddate	timestamp without time zone	End Date of Catalog Information
scsources	Character varying(255)	Sources of Catalog Information
scorigformatid	Integer	Original Format Id
scpgareapolygon	Geometry	Source Geometry Catalogue area polygon in Postgis format
scareapolygon	Character varying(5120)	Polygon in WKT String format
scpgareamultipolygon	Geometry	Source Geometry Catalogue area multipolygon in Postgis format
scareamultipolygon	Character varying(5120)	Multipolygon in WKT String format

B.46 SSOURCEMFD (Seismic Source Magnitude Frequency Distribution)

Table Name: SSOURCEMFD (Seismic Source Magnitude Frequency Distribution)

Table Description: Stores Seismic Source - Magnitude Frequency Distribution Information. Contains specific information for a seismic source with a given Magnitude Frequency Distribution. Each SEISMICSOURCE for a given magnitude frequency distribution may have 1 or more SSOURCEMFD records. SSOURCEMFD data may also be stored for discrete distributions (when ssmfddisctag=true). In the case of discrete distributions, the delta magnitude (or bin-width), number of intervals and bin values (magnitude, seismicity rate pairs in string format) are kept.

Primary Key: SEISMICSOURCE(ssid)+MAGFREQDISTN(mfdcode)
+SSOURCEMFD(ssmfdseqnum)

Foreign Key: References SEISMICSOURCE(ssid)
References MAGFREQDISTN(mfdcode)
References MAGRUPTURERELATION(mrrcode)

Name	Data Type	Comment
ssid	integer NOT NULL	Seismic Source Id (SEISMICSOURCE)
mfdcode	character(15) NOT NULL	Magnitude Frequency Distribution Code (MAGFREQDISTN)
ssmfdseqnum	integer NOT NULL	Seismic Source Magnitude Frequency Distribution

		Sequence Number
ssmfdmagnitudemax	double precision	Maximum Magnitude
ssmfdmagnitudemin	double precision	Minimum Magnitude
ssmfddeltamaggr	double precision	Delta Magnitude Gutenberg-Richter, or Bin-width
ssmfdvala	double precision	a Value, ex. 3.7256
ssmfdvalb	double precision	b value, ex. 0.7319
ssmfdlambda	double precision	Lambda Value, ex. 0.6865
ssmfdbeta	double precision	Beta Value, ex. 1.6853
ssmfdvalaorig	double precision	Original a value if source geometry catalog stores cumulative a values (note that a values in the database should be incremental)
ssmfdcharmagnitude	double precision	Characteristic Magnitude, ex. 6.87
ssmfdcharrate	double precision	Characteristic Rate, ex. 3.90195e-05
ssmfdcharmagsd	double precision	Characteristic Magnitude Standard Deviation, ex. 0.12
ssmfdtrunclevel	double precision	Characteristic Distribution - Truncation Level
ssmfdmodelweight	double precision	Model Weight, ex. 1.0
ssmfdmomentrate	double precision	Moment Rate, ex. 1.8250e+15
ssmfdstrike	integer	Seismic Source strike, ex. 0. Strike is “the fault-trace direction in decimal degrees (0 to 360, relative to North), defined so that the fault dips to the right side of the trace,” refer: www.opensha.org
ssmfddip	integer	Seismic Source Dip, ex. 50. Dip is “the angle of the fault in decimal degrees (0 to 90, relative to horizontal),” refer: www.opensha.org
ssmfdrake	integer	Seismic Source Rake, ex. 90. Rake is “the direction the hanging wall moves during rupture, measured relative to the fault strike (between -180 and 180 decimal degrees),” refer: www.opensha.org.
ssmfddipdirection	character(5)	Dip direction
ssmfddowndipwidth	double precision	Downdip Width, ex. 19.58112. Downdip width is “the average width of a fault or earthquake rupture surface measured in the down-dip direction,” refer: www.opensha.org
ssmfdtopoffault	double precision	Top of Fault, ex. 0

ssmfdfaultlength	double precision	Fault Length, ex. 52.8888
ssmfdsliptypecode	integer	Slip Type Code; 1-Normal, 2-Reverse, 3-Strike-slip
mrrcode	character(15)	Foreign key – Magnitude Rupture Relation Id (MAGRUPTURERELATION)
ssmfddisctag	boolean	Discrete tag, True if discrete, False otherwise
ssmfddeltamag	double precision	Magnitude delta for discrete distribution (or bin-width), i.e. 0.1
ssmfdnumintervals	integer	Number of intervals/bins for the discrete distribution
ssmfddiscvalsstring	character varying (2048)	Seismic source magnitude frequency distribution Values string. Contains (magnitude, seismicity rate) pair values to handle the case of a discrete magnitude frequency distribution.

APPENDIX C Hazard Database - Logic Tree Implementation

Logic trees, first introduced in probabilistic seismic hazard analysis more than 20 years ago, are considered a state-of-the art tool to quantify epistemic uncertainty, or uncertainty associated to the inputs of a calculation. Bommer and Scherbaum [4] note that one of the important considerations in setting up logic trees is to ensure that branches be both mutually exclusive and collectively exhaustive.

Figure below shows part of a logic tree illustrating the said properties. Mutually exclusive refers to the characteristic where each set of branches emanating from a node are independent of each other. Collectively exhaustive refers to the property where each set of branches emanating from a node have a sum of probabilities equal to 1.

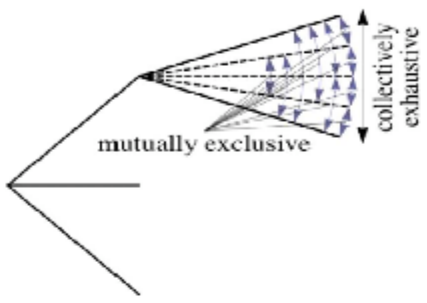


Figure C1 Schematic illustration of part of a logic-tree, showing where, in theory, the principles of mutual exclusivity and collective exhaustiveness apply to branches

C.1 Logic Trees Implementation in GEM1

GEM1 implements each logic tree as a set of all the logic tree end-paths (end-branches) for a given Hazard Input Logic Tree Calculation Model. Each of the logic tree end-paths correspond to a single hazard calculation (the branches of the logic tree end-path defines the input parameters used for the calculation). A hazard calculation results in a hazard map/curves. The resulting set of hazard calculations can then be weighted and summed to obtain the final hazard map corresponding to the full logic tree.

There are four major steps involved in generating hazard calculations from logic trees, as follows:

1. Set up the Logic Tree Structure
    - Set up Parameter Types
    - Set up Parameter Values per Parameter Type
    - Set up Parameter Type Levels of logic tree
2. Set up the Hazard Input Logic Tree Calculation Model
    - Assign Logic Tree Structure
    - Generate Logic Tree end-paths
    - Assign weights to each branch of each logic tree end-path
    - Assign a source model to each logic tree end-path
3. Calculate hazard for each logic tree end-path
4. Calculate hazard for entire logic tree

C.2 Logic Tree Usage in GEM1

Hazard Calculation for the USGS NSHM 2008 Nevada Faults Model

We consider the following logic tree taken from the USGS NSHM 2008 [5] (see Illustration E.1). For calculating hazard for Nevada, we take this as the logic tree and parse the provided input files to obtain source models to be used for hazard calculation. (Input files can be found at the USGS Website, 2008 NSHM Software, Western United States, WUS.zip)[6]

Step 1: Set up the Logic Tree Structure for Nevada faults

GEM1 stores the corresponding Logic Tree Structure as follows where the 4 parameter types corresponding to the 4 levels of the logic tree are Fault Model, Dip Uncertainty

Model, Magnitude Uncertainty Model and Ground Motion Model. The affected tables are LOGICTREESTRUC, PARAMETERTYPE, PARAMETERVALUE, LTREEPARAMTYPELEVEL.

This logic tree structure is generalized to contain the following information:

Logic Tree Structure: USGS NSHM2008 IMW Faults

Number of Levels: 4

Level 1: Parameter type: Fault Model

- Level 1 Branching:
  - Parameter Value 1: Characteristic distribution
  - Parameter Value 2: Gutenberg-Richter distribution

Level 2: Parameter type: Dip-Uncertainty Model

- Level 2 Branching:
  - Parameter Value 1: 40 degrees
  - Parameter Value 2: 50 degrees
  - Parameter Value 3: 60 degrees

Level 3: Parameter type: Magnitude-Uncertainty Model

- Level 3 Branching:
  - Parameter Value 1: Subtract 0.2 - coded MagDeduct02
  - Parameter Value 2: No Change in Minimum Mag -coded MagNoChange
  - Parameter Value 3: Add 0.2 to Minimum Magnitude – coded MagAdd02

Level 4: Parameter Type: Ground Motion Model

- Level 4 Branching:
  - Parameter Value 1: Boore and Atkinson (2008) - coded BOAT2008
  - Parameter Value 2: Campbell and Bozorgnia (2008) – coded CABO2008
  - Parameter Value 3: Chiou and Youngs (2008) – coded CHYO2008

Step 2: Set up the Hazard Calculation Logic Tree Input Model

Once the logic tree structure is set up, the Hazard Calculation Logic tree Input Model is then set up. First, the appropriate logic tree structure is assigned. Then, the logic tree end-paths are generated based on the chosen logic tree structure.

For the Nevada Model, there are 2 x 3 x 3 x 3 = 54 logic tree end paths that are generated from the given logic tree structure (derived from the different combinations of branching per level).

Next, weights are assigned to each branch of each logic tree end-path. Each branch will thus contain a parameter type_parameter value_branch weight triple as shown in the following illustration of how end-paths are stored in tables:

Logic Tree end-path 1:

FaultModel_CHAR_0667.DipUncertModel_40deg_02.MagUncertModel_MagDeduct02_02.GrdMotionModel_BOAT2008_0333

Logic Tree end-path 2:

FaultModel_CHAR_0667.DipUncertModel_40deg_02.MagUncertModel_MagNoChange_06.GrdMotionModel_CABO2008_0333

Logic Tree end-path 3:

Faultmodel_CHAR_0667.DipUncertModel_40deg_02.MagUncertModel_MagAdd02_02.GrdMotionModel_CHYO2008_0334

… up to 54 logic tree end-paths.

Finally, a source model is assigned to each logic tree end-path (as previously parsed from input files).

Step 3: Calculate hazard for each logic tree end-path

In this step, the input parameters defined from the logic tree end-path is used for a hazard calculation.

Step 4: Calculate hazard for entire logic tree

In this step, the set of all the hazard calculations resulting from all the logic tree end-paths are summed and weighted to obtain the final hazard map corresponding to the full logic tree.

APPENDIX D Sequence Diagrams

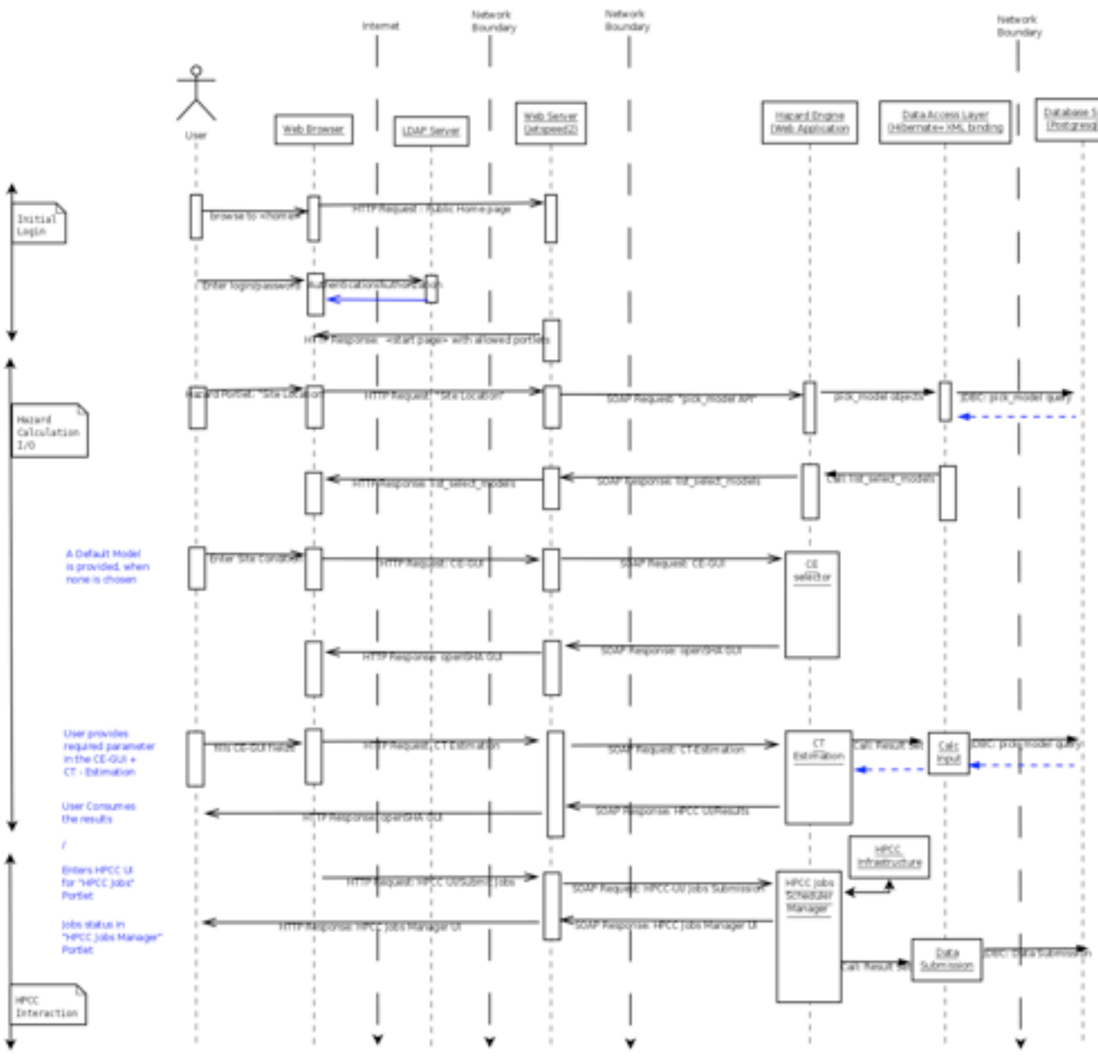

Internet
Network Boundary
Network Boundary
Network Boundary
User
Web Browser
LDAP Server
Web Server (Jetspeed2)
Hazard Engine (Web Application
Data Access Layer (Hibernate+ XML binding
Database Server (Postgresql 8.3)
Initial Login
browse to <home>
HTTP Request : Public Home page
Enter login/password
Authentication/Authorization
HTTP Response: <start page> with allowed portlets
Hazard Calculation I/O
Hazard Portlet: "Site Location"
HTTP Request: "Site Location"
SOAP Request: "pick_model API"
pick_model objects
JDBC: pick_model query
HTTP Response: list_select_models
SOAP Response: list_select_models
Call: list_select_models
A Default Model is provided, when none is chosen
Enter Site Condition
HTTP Request: CE-GUI
SOAP Request: CE-GUI
CE selector
HTTP Response: openSHA GUI
SOAP Response: openSHA GUI
User provides required parameter in the CE-GUI + CT - Estimation
fills CE-GUI Fields
HTTP Request: CT Estimation
SOAP Request: CT-Estimation
CT Estimation
Call: Result Set
Calc Input
JDBC: pick_model query
User Consumes the results
HTTP Response: openSHA GUI
SOAP Response: HPCC UI/Results
/
Enters HPCC UI for "HPCC Jobs" Portlet
HTTP Request: HPCC UI/Submit Jobs
SOAP Request: HPCC-UI Jobs Submission
HPCC Infrastructure
HPCC Jobs Scheduler Manager
Jobs status in "HPCC Jobs Manager" Portlet
HTTP Response: HPCC Jobs Manager UI
SOAP Response: HPCC Jobs Manager UI
Call: Result Set
Data Submission
JDBC: Data Submission
HPCC Interaction

Figure D1 Sequence Diagram for a typical Hazard Calculation