



Development

ISPP Aggregator High Level Design

Draft 1.0, 10th September 2008

Document History

<i>Date</i>	<i>Version</i>	<i>Description</i>	<i>Author</i>
10 th September 2008	1.0 Draft	Created	Ian Ibbotson

Table of Contents

1 INTRODUCTION.....	3
2 SYSTEM OVERVIEW.....	3
3 DESIGN CONSIDERATIONS.....	3
3.1.1.1 Assumptions and Dependencies.....	3
General Constraints.....	3
Goals and Guidelines.....	4
3.1.1.2 Development Methods.....	4
4 ARCHITECTURAL STRATEGIES.....	4
5 SYSTEM ARCHITECTURE.....	6
5.1 DATABASE SERVER COMPONENTS.....	7
5.1.1 SOLR Master Server	7
5.1.2 SQL Server.....	7
5.2 FRONT END COMPONENTS.....	7
5.2.1 Deposit SPI.....	7
5.2.2 Retrieval SPI.....	7
5.2.3 Indexing SPI.....	7
5.2.4 Data Provider Portal Webapp.....	7
5.2.5 SOAP Services Endpoint.....	7
5.2.6 SOLR Slave.....	8
5.2.7 SRU Server.....	8
5.2.8 ISPP End User Application.....	8
6 DEPLOYMENT MODELS.....	8
7 GLOSSARY.....	8
8 BIBLIOGRAPHY.....	8

1 Introduction

This document presents a high level design for the architecture of the ISPP aggregator service. It's intended to be read by development and technical staff implementing the ISPP aggregator and provides a high level map for the aggregator subsystems.

2 System Overview

The ISPP Aggregator is a component based solution which exposes deposit, retrieval and search services to a range of service provider peers.

The software system itself is split into separate web application components which address separate concerns. Specifically, these relate to the services provided to record uploaders, search services, and end user applications.

Each front end web application is a relatively thin binding layer onto a deeper system interface which actually provides the required service. This ensures the architecture is kept clean and not tied to any specific front end protocol or specification.

The overall aim of the design is to meet the functional and non functional requirements of the ISPP project. Specifically however, the architecture separates the performance requirements for data providers from those for end user searching. Effectively removing any dependency between the two. This is achieved through the use of a core master index for the search service who's only task is to collate incoming updates and distribute copies to listening slaves. Essentially this master index is the pivot point between uploaded data and users being able to locate records.

The system design tries where possible to take account of current best practise in the areas of metadata and document aggregation and search. Specifically the design reflects a trend towards lightweight restful services, whilst providing the more traditional SOAP based interfaces.

3 Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

3.1 *Assumptions and Dependencies*

3.1.1 Hardware & OS Dependencies

The solution is required to run on Windows NT 2003 server and be supported by a Microsoft SQLServer 2003 database. The web application container is tomcat and the deployed system will require a java 6 runtime.

3.1.2 End user characteristics

A number of performance metrics are defined ... reference.

ISPP does not itself present any end user interfaces (Save the Data Provider Portal) however, the applications and interfaces ISPP supports must be accessible to the required standard.

3.1.3 Possible / Probable changes

- Work is still ongoing with document schemas both in terms of data content and relationships to vocabulary management services. The system will need to be able to cope with changes to schemas and terminology lists.

3.2 General Constraints

- Standards Compliance
- Interoperability Requirements – Interoperability is a critical and key aspect of the project. All data providers, and all service consumers need to be able to interact with the ISPP system without problems. Critical care needs to be taken over the formation of interfaces and specifications being mindful of the range of development environments likely to be in use by client systems. This will avoid unwanted interoperability issues caused by, for example, different implementations of SOAP/WSDL standards.
- Interface and Protocol Requirements
- Security Requirements
- Memory and Other Capacity Limitations
- Performance Requirements
- Network Communications
- Verification and Validation Requirements

3.3 Goals and Guidelines

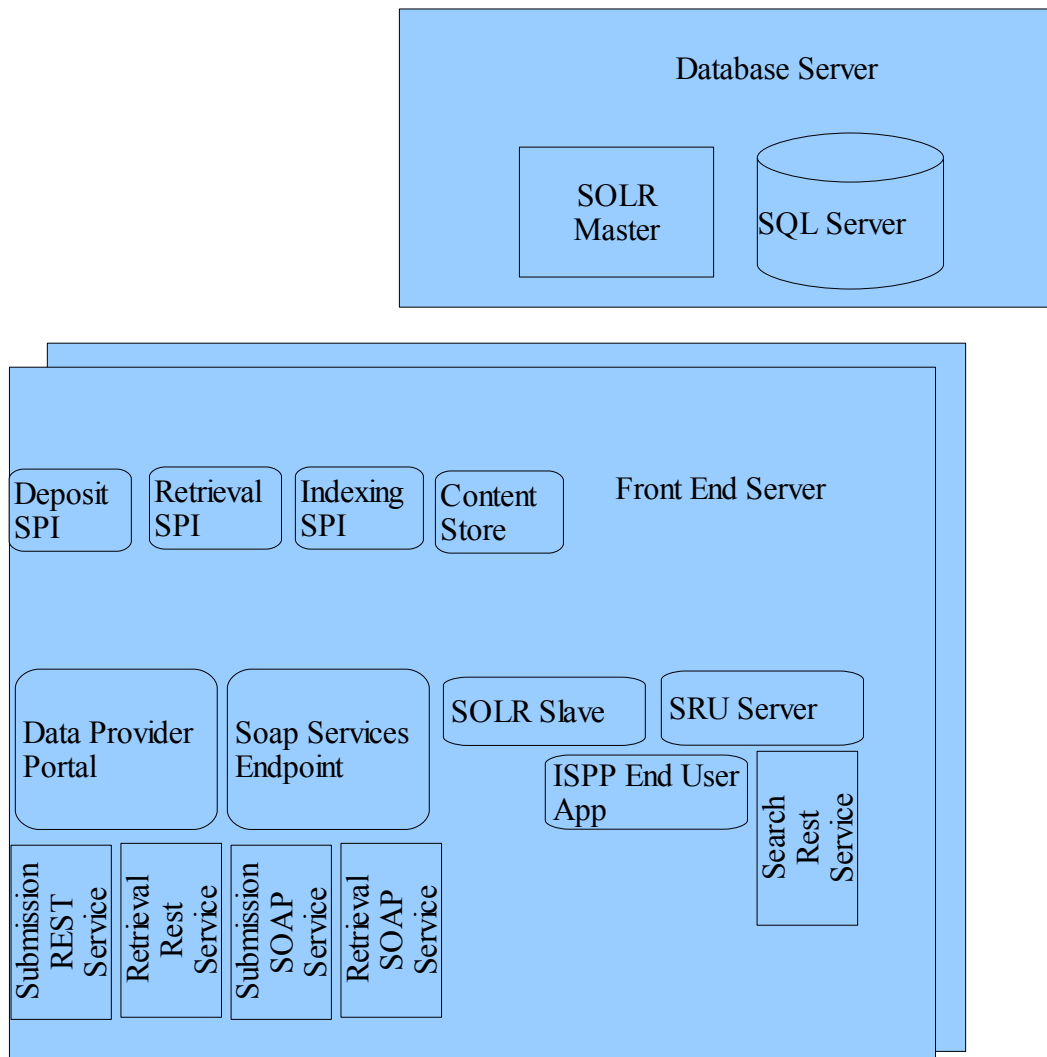
- The ultimate goal is to put into the hands of users an apparently simple “Google-like” search service for information covering the range of resources supported by the ISPP project which delivers relevant and precise information. The goal is to create a service than can be bound into a number of target environments and tuned to the specific needs of that service, whilst still providing an apparently simple free text / place based search.
- Performance is considered a key requirement and borne out in the system metrics.
- The system architectural tries to adhere to the KISS principle and keep component interfaces clean, each one doing 1 job well.

3.4 Development Methods

The system is being developed using an agile / iterative deployment methodology.

4 Architectural Strategies

5 System Architecture



5.1 Database Server Components

The following components are located on a central master server.

5.1.1 SOLR Master Server

So that all LA's submitting data to the ISPP system are not impacted by the activity of live user searching the SOLR master server provides a central indexing point for all incoming records. This solr instance is only used to process updates and distribute indexes out to slave solr indexes. It is never used to actively search the ISPP collection. This arrangement also makes it easy to add new servers to the front end cluster.

5.1.2 SQL Server

The SQL Server engine is used for all non information-retrieval storage, from copies of incoming documents to administrative and user account data.

5.2 Front End Components

Front end components are deployed on each running web server in the ISPP cluster.

5.2.1 Deposit SPI

The Deposit System Programming Interface is responsible for presenting a deposit service to all front end web applications. It's the fundamental implementation which accepts and validates incoming ECD/FSD/Other documents, and coordinates the activity of the indexing, database and content store.

5.2.2 Retrieval SPI

So that LA's may retrieve records submitted the retrieval SPI presents services for listing and pagination of records from specific date ranges or other criteria.

5.2.3 Indexing SPI

The Indexing SPI is the primary interface to the SOLR master server. It's responsible for all transactions with the searchable index.

5.2.4 Data Provider Portal Webapp

The Data Provider Portal is the primary web interface used by LA's and other data providers to manage their interaction with the ISPP. It presents restful bindings to the deposit and retrieval SPIs as well as utility and administrative functions to ISPP staff.

5.2.5 SOAP Services Endpoint

The SOAP Services endpoint presents the ISPP SOAP services which are then bound to the SPI implementations.

5.2.6 SOLR Slave

Each SOLR Slave is updated from the SOLR master to decouple indexing and search activity in the system.

5.2.7 SRU Server

The SRU Server presents the primary interface for google-like searching of the ISPP application.

5.2.8 ISPP End User Application

Is a platform for presenting a fully functional end user application.

6 Deployment Models

The components on section 2 have been arranged according to current estimates and plans. However, additional vertical and horizontal scalability is available by breaking out the SPI components into their own servers.

7 Glossary

8 Bibliography