CANADIAN GEOSPATIAL DATA INFRASTRUCTURE

ARCHITECTURE DESCRIPTION

Revision: Version 1
December 11, 2001

Prepared by the

CGDI Architecture Working Group

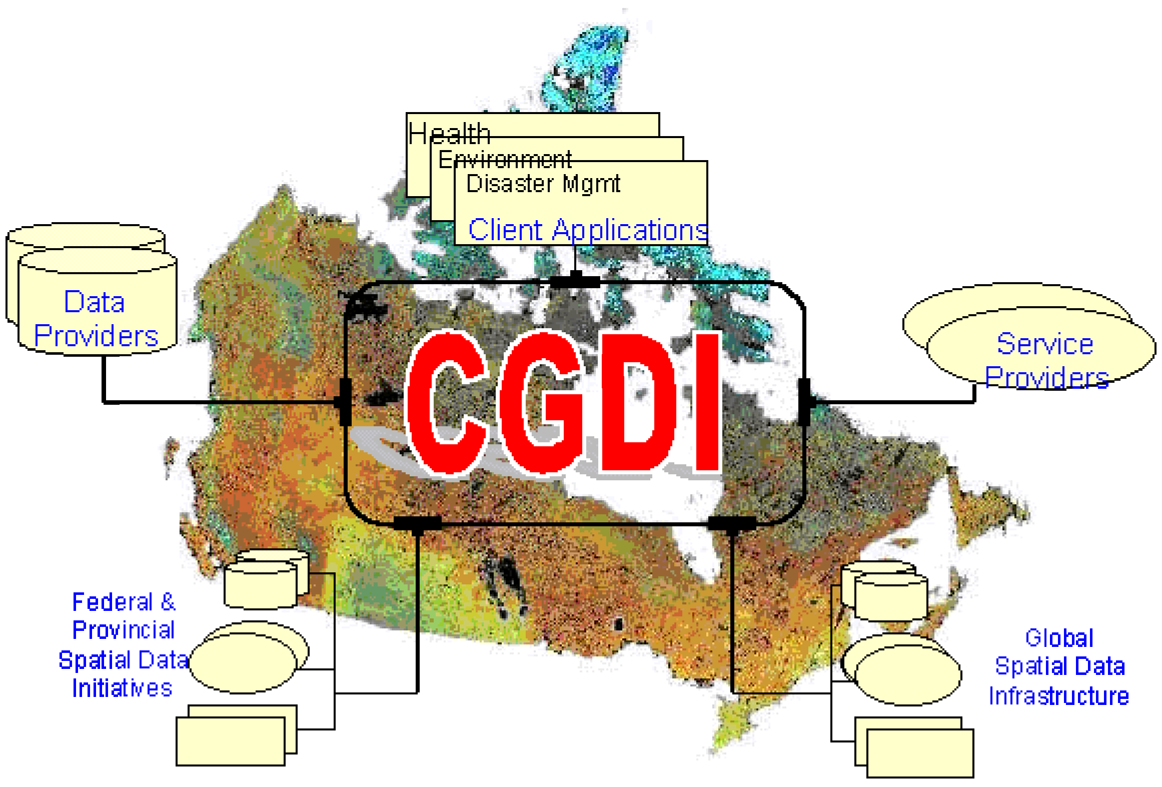

Table of Contents

1 INTRODUCTION

The Canadian Geospatial Data Infrastructure (CGDI) is a distributed set of data, as well as services and applications that enable the sharing and use of geospatially referenced information. The CGDI is being developed by the Geoconnections program. A complete introduction to the CGDI and its various aspects is found in the CGDI Target Vision.

CGDI is an open information technology infrastructure that is based upon publicly available specifications. The architecture is designed to enable the implementation of systems to support service providers, data providers and application developers, using interoperable and re-usable components. This goal is achieved largely through specifying the interfaces of these services. These specifications draw upon the International Organization for Standardization (ISO) 19100 series of abstract standards for Geographic Information, and related implementation specifications under development by the Open GIS Consortium (OGC).

This architecture description document is one of a trio of evolving documents that describe the CGDI:

1. The CGDI Target Vision,
2. The CGDI Architecture Description, and
3. The CGDI Implementation Plan.

1.1 Intended Audience

This document is intended to introduce the CGDI Architecture to data providers, service providers, and application developers so that they can gain an overall understanding of the services and related aspects of the CGDI.

1.2 Scope

This document provides a high level overview of the architecture of the CGDI. It describes the range of services that comprise the CGDI, and provides context and reference information for the more technical documents that describe the individual services and other aspects of the architecture. It also describes the underlying architecture that is common to all services.

2 A CONCEPTUAL OVERVIEW OF THE CGDI

2.1 *Characteristics of the CGDI*

The CGDI vision indicates that the architecture of the CGDI will:

a) enable universal access to any kind of geospatial information, anywhere, anytime;

b) enable applications to discover and access remote online information through a distributed infrastructure;

c) enable integration of disparate geospatial information to provide seamless views;

d) enable the seamless chaining of applications, data and services or combinations of these;

e) provide geospatial update and exchange capabilities, enabling collaborative activities;

f) promote the sharing of geospatial semantics to make integration of information easier;

g) enable wide-scale interoperability by adhering to common and open information standards and specifications;

h) facilitate the development of effective partnerships with regional and sector-specific Spatial Data Infrastructures (SDIs), and linkages with other national SDIs to form a Global Spatial Data Infrastructure (GSDI);

The conceptual architecture for CGDI includes three main elements: 1) data; 2) services; and 3) applications. The diagram in Figure 1 diagram illustrates these elements.

The organization of the CGDI shown in Figure 1 consists of a network of providers of information and services, and consumers or users with applications that use the information and services provided by those organizations.

The infrastructure will provide clients with access to information and services, and will enable client applications to make use of geospatial information. The CGDI builds upon federal, provincial, territorial, municipal, and industrial spatial data and initiatives, and is a collaborative information framework that integrates and functions with other spatial data infrastructures in existence around the world.

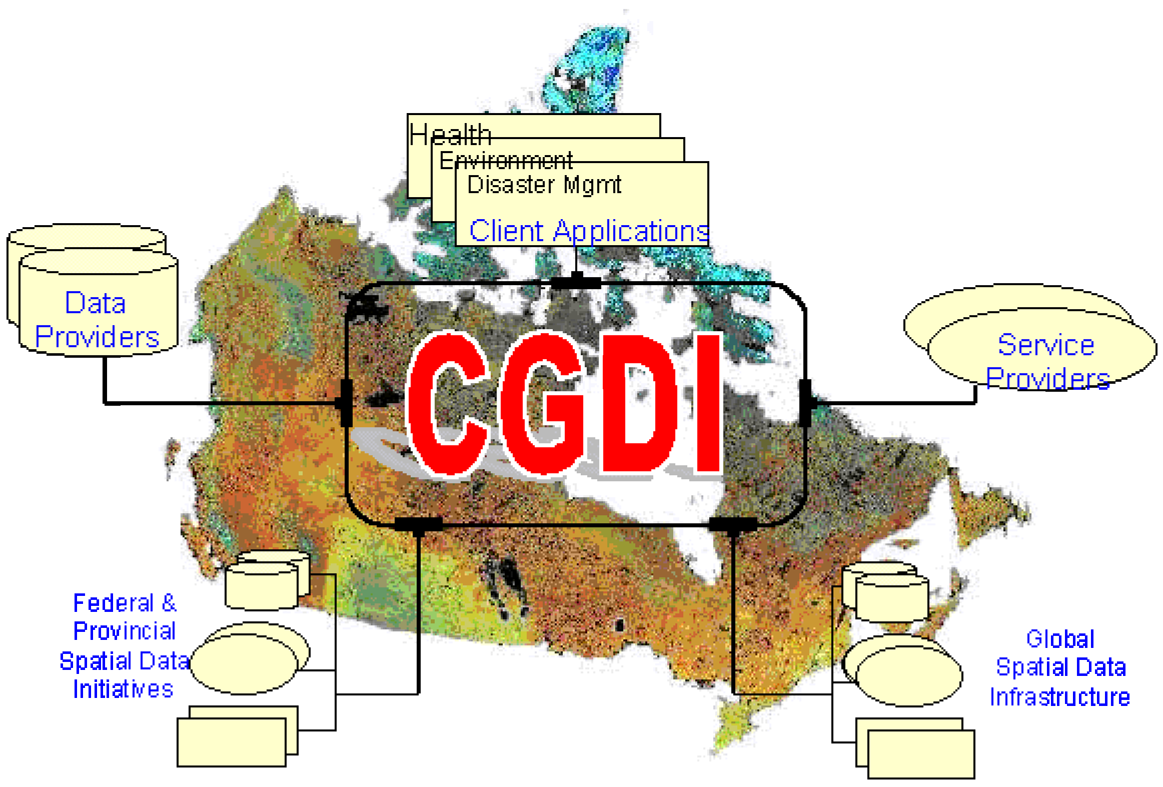

Figure 1: CGDI Conceptual Architecture

The remainder of this section expands upon this generalized conceptual description of the CGDI. Geospatial objects are described, as are the services and related aspects of the infrastructure that allows a user to interact with those geospatial objects.

2.2 Geospatial Objects

Geospatial objects describe real world entities that are employed in GIS applications. They describe geographic features, geographic measurements, spatial reference systems, geographic transformations, etc.

Figure 2 shows a representative set of fundamental geospatial objects. Each high level object is represented as a package containing a collection of related objects, which, when implemented, may be distributed across the infrastructure. Some are at the client level for visualization, some are at the application server level for processing, and some are at the database or repository level for storage and access.

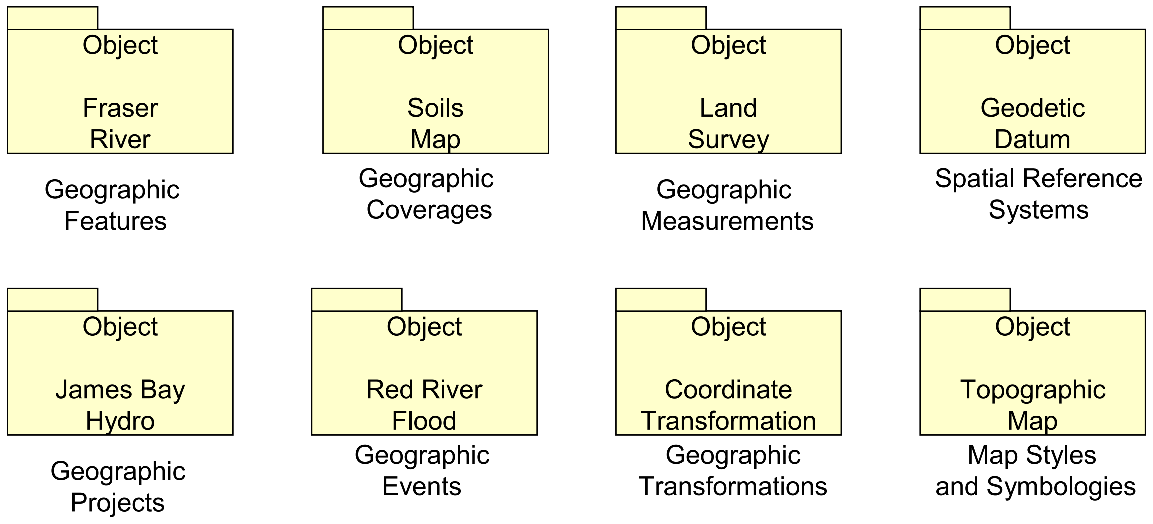

Figure 2: Geospatial Objects for CGDI

Typically, one can expect to find the data associated with these objects in various types of repositories, or Spatial Object Databases. These include:

- Feature Databases, i.e., geographic feature servers;
- Coverage Databases, including imagery and other types of spatial distributions;
- Map Style Libraries;
- Geographic Event Databases;
- Map Symbol Libraries;
- Geographic Measurement Databases, e.g., surveys, hydrographic and oceanographic measurements, and;
- Spatial Reference System Dictionaries.

These geospatial objects are examples of elements of the infrastructure that would be utilized by many client applications. The purpose of CGDI is to make the interfaces to these objects available to clients and providers with seamless views of the information.

2.3 Servers and Services

The CGDI defines a set of abstract services that enable access to geospatially-referenced information, and a set of interfaces to these services. One or more of these services is invoked by every interaction that a user has with CGDI. Users will access most of these services through the help of client software applications. The visible part to client software applications is comprised of well-known and "open" sets of software interfaces.

Definitions of component, interface, and service are logically consistent with those found in the ISO TC211 19119 document on Geographic Information – Services which describes key elements of the architecture:

> **Service**: A collection of operations, accessible through one or more interfaces, that allows a user to evoke a behavior of value to that user.
>
> **Interface**: A specification for a set of operations that are made externally available by a component to other components.
>
> **Component**: Software that packages the implementation of a service and provides the realization of a set of interfaces. Components can be installed on multiple computers.
>
> **Server:** A physical installation of a component that delivers a service. Servers are implemented using a particular Distributed Computing Platform (DCP).
>
> **System:** Servers and data organized to accomplish a goal.

2.3.1 Examples of Services

CGDI services include things like:

- Services to assist in the discovery and direct access of services and information;
- Web Map Service, to generate rendered maps from feature data stores using Web Feature Services;
- Web Feature Service, to support retrieval or editing of individual geo-spatial features and their properties over the Internet from any data stores;
- Web Coverage Service, to provide delivery of data coverages such as digital elevation data and other fixed or variable sized matrix data;
- Map Styling Service, and services to access Map Symbol Libraries, to support styling of geographic features in an encoding form parsable by a Web Map Service;

- Event Notification Service, to notify applications of changes to reference datasets or other services.

2.3.2 Examples of Components

A few types of CGDI compliant components are described below.

- **Mapserver**: free software that complies with the CGDI Web Map Service specification, and can be used to set up a Web Map Server
- **MetaManager**: software that complies with the GeoData Discovery Service specification, and can be used to publish metadata

2.3.3 Examples of Servers

A few types of CGDI servers are described below.

- **Web Map Server**: a server that delivers symbolized graphics (in one form or another) across a common interface to a viewer client. The Web Map Server may also provide an interface for associating default symbology with a dataset for use by client applications.
- **Web Feature Server**: a server that provides access to data across a common set of interfaces (typically as the result of a query) to any client. Interfaces supported by a Web Feature Server may include the capability to create a feature, delete a feature, insert a feature and lock a feature using some filtering mechanism.
- **Geodata Discovery Server** – a server maintained by a registration authority that describes registered types of services or definitions of geospatial object types.

2.4 Type and Instance Registries

Registries are used to keep track of objects in the CGDI. A number of services interact with registries, either to populate them, or to access the information that they contain. There are two kinds of registries:

Type Registry: A listing of the different types (classes) of objects, such as services, components, or events, which are recognized by CGDI services or applications.

Instance Registry: A listing of the individual services, components, datasets, or other things that comprise the CGDI or are relevant to its users. Registries are used to identify, locate, and describe individual instances.

Registry services provide descriptive information concerning CGDI objects, including:

- Metadata about geospatial datasets that allow clients to locate geospatial datasets;

- Map styles and map symbols that allow map servers and viewer clients to render features in standard ways (such as are found on geological maps and nautical charts);
- CGDI services;
- CGDI compliant applications.

The information included in the registries describe significant characteristics of the resource such as its purpose, content, spatial and temporal coverage, cost, and where it can be located.

The CGDI architecture assumes there will be many registries within the CGDI. Some registries may be built by searching other registries, or designed in such a way to provide distributed search capabilities of peer registries.

2.5 Data and Framework Data

The primary motivating factor behind the development of the CGDI is the series of problems encountered when attempting to access and use geospatial data. Naturally, CGDI facilitates the use of and access to all geospatial data. However, it also promotes sharing and compatibility of geospatial data by identifying a common set of Framework Data.

Framework Data provides a common geographical reference for the country. It is widely used and underpins most geospatial applications. Framework Data includes physical features such as roads and rivers, as well as conceptual boundaries such as municipal and provincial boundaries, and spatial alignment features. By identifying and providing access to this common set of Framework Data, CGDI will facilitate the referencing and integration of all geospatial data. For more information about Framework Data, see the CGDI Framework Data Definition.

The CGDI will provide access to the CGDI framework data itself, from servers provided by participating agencies. In practice, the need for framework data will vary from one community to another, and data that is considered CGDI framework data may not meet all the framework data needs for specific user communities. For this reason, the architecture services will support both the CGDI framework datasets (like Federal Electoral Districts) as well as datasets that are designated as “framework” by individual user communities.

Of great significance in the notion of Framework Data is data that can be geolinked to Framework Data, since Framework Data is regarded as data that is used for referencing other data. For example, a criminologist may wish to look at crime data versus school districts, while a health professional may be interested in the incidence of VR resistant bacteria in various health districts, or even in school districts or counties. In many cases, to establish spatial relationships with framework data, the user is not required to copy the framework data in question (e.g. county boundaries), but merely to establish a relationship between their own data and the desired framework data elements. This can be achieved through the introduction of geodata relationships that point to framework data elements from the user's data. Multiple types of relationships can be constructed in this manner allowing some organizations to focus on information content, while leaving the description of spatial location to others.

2.6 Client Applications

Beyond the generic capability provided by the CGDI services, CGDI supports a rich set of client software applications, which are built to satisfy specific requirements in a wide variety of application domains. A **client** is a software component (an application) that uses one or more services.

The intent of this section is not to make an exhaustive list of all possible software applications that may access the CGDI, but to indicate that such software components are developed using some of the server components presented above. Client software applications described below are used by CGDI users and represent a minimal list derived from on-going CGDI activities. It corresponds to requirements derived from roles played by current identified communities and users of CGDI. While some client software applications such as a "Discovery Client" or a "Viewer Client" support generic requirements, others like a "Publisher Client" exist to satisfy the requirements of a Data Provider community and its desire to efficiently distribute its data to all clients of its own community. These are further described below.

- **Viewer Client**: a component that renders (or possibly simply displays) graphics that come from map server components (includes layer control capability, fusion of multiple layers, interface for locating datasets of interest, etc.). A viewer client may include a gazetteer to provide a mapping from place name to a given geometry. A viewer client may also support transformation between location reference systems (e.g. from postal codes to mobile phone cells). A Viewer Client requests geodata through a Map Server. While a typical Map Server renders the data and generates an image, a Map Server may apply map styles directly to the data during the rendering operation.
- **Editor Client**: a component that allows updating information in a data store or adding new information to a data store. An Editor Client interacts with a Data Server and can add or update geospatial data in a repository. It also may provide the ability to construct relationships among items in one or more repositories.
- **Discovery Client**: A component for querying metadata, selecting a map or data service, and adding that service or data to another application such as a viewer client. The query screen might have three parts in which text, keyword, and geographic constraints can be defined. The Discovery Client would normally search the contents of an instance registry to find the services and datasets of interest.

A rich set of client software applications based on core CGDI components, interfaces and services, built by CGDI developers and providers, will ultimately deliver much of the anticipated benefits to all Canadians.

2.7 Robustness and Redundancy

Some of the services and registries in the CGDI are critical to the ongoing operations of the Infrastructure. For example, the Service Registry Service keeps track of where CGDI services can be found. If this service was to be unavailable, client applications would not be able to locate the services that they require in order to complete a task. The services that support access to these critical registries are considered to be "core" services.

Some core services will be implemented by identifying a "Master" implementation that is mirrored to other identical servers, but most services will be implemented on multiple systems for one or more of the following reasons:

- Robustness arising from redundancy (so that portions of the architecture will continue to work while others are out of service or inaccessible).
- efficiencies that may be obtained by coupling local datasets with local service implementations (so that custodians of information can provide access to that information from their own systems), and
- benefits arising from local management and maintenance of service implementations.

2.8 Security and Authentication

Security and authentication has not been addressed by current efforts to develop the CGDI Technical Architecture, but these services are critical for global geospatial data processing and operations. Even a simple service such as rendering a map layer by a Web Map Server has security implications. Different sources of geographic data may have different levels of sensitivity in terms of who is allowed to see them. The need for security and authentication mechanisms increases with the need to share information in an open and interoperable fashion, particularly in those operations that create or update data.

CGDI will not re-invent new security technology and related interfaces, but will instead use available standards as much as possible and promote extension of these standards, if required.

3 ARCHITECTURAL REQUIREMENTS

3.1 Architectural Context for the CGDI

The CGDI is a spatial information technology infrastructure. The high-level context diagram below illustrates the fact that CGDI is driven by a variety of key business applications from both the private and public sector. Implementation of the CGDI will offer these enterprise applications a set of software components, services and data that will reduce the difficulty, cost and time of application development and provide value that can come only from a network of clients and providers.

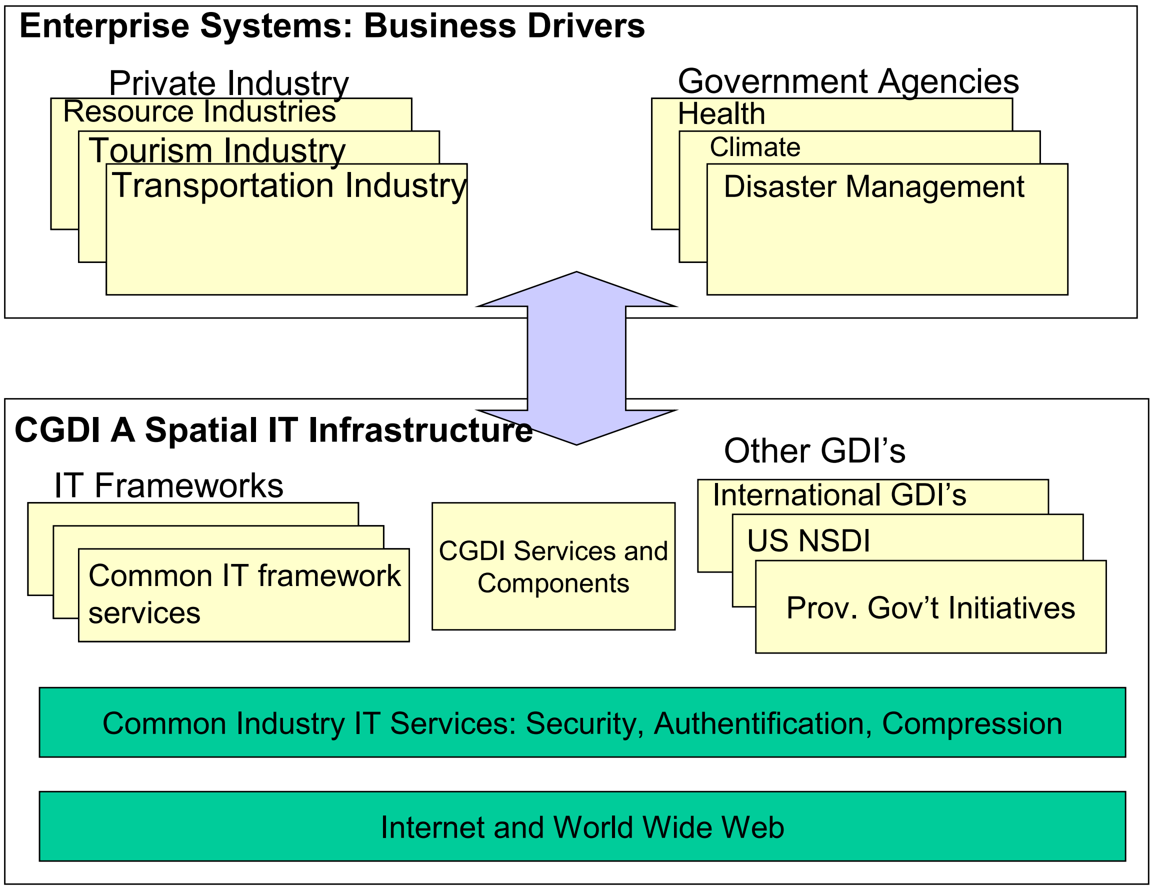
Enterprise Systems: Business Drivers
Private Industry
Resource Industries
Tourism Industry
Transportation Industry
Government Agencies
Health
Climate
Disaster Management
CGDI A Spatial IT Infrastructure
IT Frameworks
Common IT framework services
CGDI Services and Components
Other GDI's
International GDI's
US NSDI
Prov. Gov't Initiatives
Common Industry IT Services: Security, Authentification, Compression
Internet and World Wide Web

Figure 3: Information Technology Context for CGDI

A basic principle of the CGDI is that it is built on existing communications and IT infrastructure such as the Internet and the World Wide Web. It makes use of a variety of existing industry services and technologies for data security, data compression and user authentication. This approach is in line with the ISO 19100 series of standards for geographic information, and in particular, *ISO*

19101 Geographic information – Reference Model. This series of standards and their relationship to CGDI is summarized in Appendix 1.

3.2 Architectural Requirements

The principal requirements for CGDI are architectural in nature:

1. Enable online access to a wide range of geographic information and services;
2. Enable integration of geographically distributed geospatial information;
3. Enable collaboration by multilateral information exchange and synchronization;
4. Allow autonomous organizations to develop interdependent relationships in a distributed environment, and;
5. Facilitate the definition and sharing of geospatial semantics;

Each of these fundamental features is key to the success of the CGDI, yet they are enabling features rather than functional features. They are achieved through organization and structure rather than through individual component capabilities.

3.2.1 The need for Interoperability

Common to each of the requirements stated above is the need for interoperability. Interoperability is the ability of a system or component of a system to to access a variety of heterogeneous resources by means of a single, unchanging operational interface. Interoperability facilitates information sharing, and provides the freedom to mix and match information system components without compromising overall success. Interoperability allows users to:

- Find information, services, and applications, when they are needed, independent of physical location;
- Understand and employ the discovered information and tools, no matter what platform supports them, whether local or remote;
- Evolve a processing environment for commercial use without being constrained to a single vendor's offerings.

3.2.2 Aspects of Interoperability

Interoperability between systems and components of systems has several aspects:

- *Network Protocol Interoperability* allows basic communications between components.
- *Standard Interface Specifications* allow client applications to execute procedures on remote systems.

- *Data Transport Interoperability* allows transparent access to data, the sharing of spatial databases and other services regardless of the proprietary data storage format.
- *Semantic Interoperability* refers to applications interpreting data consistently in the same manner in order to provide the intended representation of the data.

Understanding of these underlying architectural requirements underscores the principle of building a geospatial data infrastructure using standards-based re-useable and replaceable components on a foundation of standard Information Technology infrastructure.

4 ARCHITECTURE DEVELOPMENT STRATEGY

The Geoconnections program is defining the CGDI architecture primarily through the work of the CGDI Architecture Working Group (CAWG). The architecture is built as much as possible on the current and ongoing work of international standards organizations. A number of other Geoconnections activities also contribute to the development of the CGDI architecture, but primarily from domain-specific or technology-specific perspectives.

4.1 Overall Approach

The International Organization for standards (ISO) is developing the 19100 series of standards for distributed geospatial data processing, based on concepts from the IEC Open Distributed Processing Reference Model (RM-ODP) described in ISO/IEC 1076-1:1995. The RM-ODP framework recognizes that architectures are complex and cannot be described in a single representation. RM-ODP presents an architectural framework in terms of five viewpoints: enterprise, information, computational, engineering, and technology. The viewpoints cover a range of issues from organizational through to the selection of technologies, in a carefully organized and hierarchical manner.

Notwithstanding the value of the RM-ODP approach, when an architecture is comprised of standards-based services it is possible to describe that architecture very efficiently by specifying the action and interface for each of the services. The pre-requisite here is that the behavior of the services and the required interfaces between those services are clearly understood.

In the case of the CGDI the nucleus of standards-based services does exist:

1. Searching Registries and Spatial Databases: *FGDC GEO/Z39.50 Search Protocol;*
2. Rendered maps: *OGC Web Mapping Service (WMS);*
3. Geographic features: *OGC Web Feature Service (WFS);*
4. Registry data: *OGC Catalogue Interface Implementation Specification;*
5. Geospatial metadata: *FGDC Content Standard for Digital Geospatial Metadata (CSDGM);*
6. Geographic Feature Encoding: *OGC Geography Markup Language.*

Developing a complete RM-ODP description of a distributed architecture would be an extensive and time-consuming activity requiring the concentrated efforts of a team of architects and engineers. The CAWG, at the workshop of 28 and 29 November 2000 concluded that it was not practical to implement a complete architectural definition for the CGDI, and to instead adopt a spiral development approach to designing the CGDI architecture, based on the specification of standards-based services.

This architectural description of the CGDI is founded on the assumption that enough domain knowledge exists to start the development of core services and the implementation of systems based on those core services. As the infrastructure continues to develop, an increasing number of services and their relationships will be specified, and existing services will be refined.

4.2 Creating Specifications for Software Interfaces

Software interfaces constitute key elements of the CGDI. Software interfaces are the glue used by a client component to invoke a service and by a server component to invoke itself or another service. CGDI will focus on identifying and promoting existing open and interoperable software interfaces, and developing new ones as required though the help of existing public standards organizations.

4.2.1 The Open GIS Consortium (OGC)

Since 1999, the Open GIS Consortium has been developing open Web-based mapping and related specifications in cooperation with a large community of public organizations and software vendors. OGC specifications allow software vendors to implement their products using open interoperable interfaces and provide end-users such as CGDI users with a larger pool of interoperable Web-based tools for geodata access and related geoprocessing services.

The OGC has developed a number of software interface specifications, including the:

- Web Map Server interface specification
- Web Feature Server interface specification
- Geography Markup Language specification

The CGDI is built around OGC specifications wherever possible, and OGC specifications development is supported by Geoconnections. More information on OGC Open Web based interface specifications can be obtained from the OGC web site (http://www.opengis.org/specifications/).

4.2.2 The International Organization for Standards (ISO)

The ISO is developing a series of abstract standards through Technical Committee 211 (TC211). The 19000 series of standards has been referred to earlier in this document, and will undoubtedly continue to play a significant role in shaping the development of the CGDI architecture. Fortunately, ISO TC211 and the OGC work very closely together, and it is anticipated that many of the specifications developed by OGC will become ISO standards. For more information about ISO, see http://www.iso.ch/iso/en/ISOOnline.frontpage

4.2.3 The CGDI Architecture Working Group (CAWG)

The CAWG consists of interested Canadian experts that are participating in the Geoconnections program. The CAWG identifies the range of services that comprise the CGDI, and identifies appropriate service specifications or the need for development of these specifications. The Geoconnections Technology Advisory Panel (TAP) endorses Service specifications that are ready to be implemented as part of the CGDI.

4.3 Describing the Architecture

The CGDI has multiple stakeholders: client organizations, suppliers, and developers, which interact with each other across distributed facilities. The architecture of the infrastructure and its supported capabilities needs to be clearly described and available to those who interact with it. Descriptions of the architecture from different perspectives are used to enable that essential communication between the diverse stakeholders: between users, architects, systems engineers and developers.

Three different perspectives of the CGDI architecture are illustrated in Figure 4. The major components are the:

Operational Architecture: which describes operational concepts and defines how operational requirements of the enterprise client and users are realized.

Technical Architecture: standards, profiles, and specifications that govern and constrain the design of components that must interoperate in the system environment.

Systems Architecture: design and implementation descriptions of components and systems.

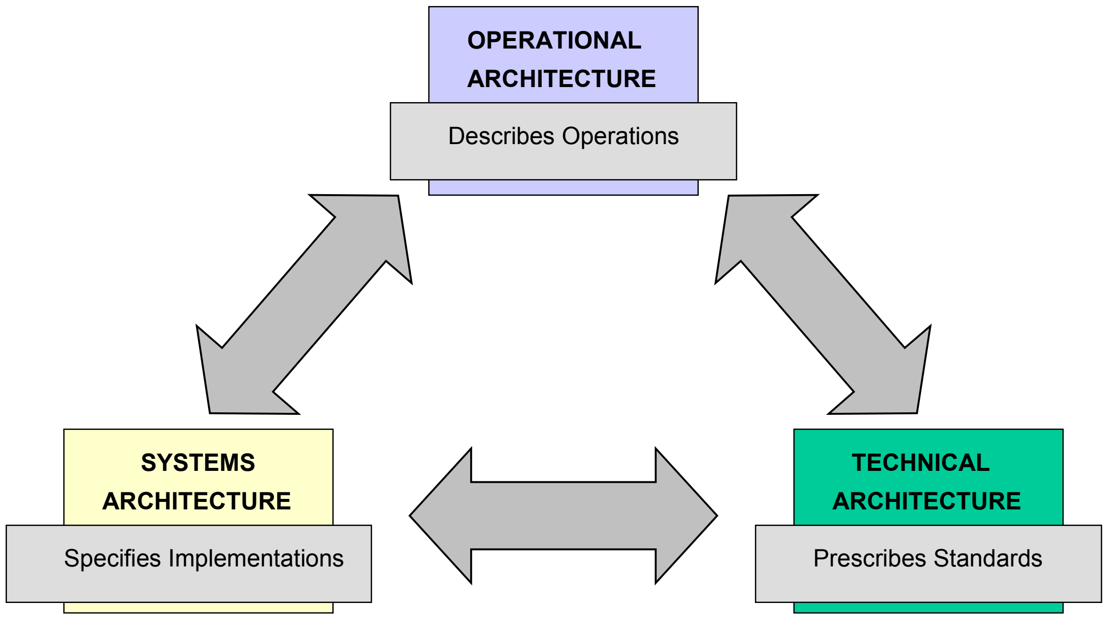

Figure 4: Different Perspectives of the Architecture

The *Operational Architecture* provides a view from the perspective of users in the language of the business enterprise or application domain. It includes use cases that describe the operational functions of the infrastructure.

The *Technical Architecture* provides a view of the infrastructure from the perspective of the architect concerned with interoperability, adaptability, maintainability, availability and quality of service.

The *System Architecture* provides a view of the infrastructure from the perspective of the individual systems engineers and developers who design, integrate, test and maintain components and operational systems.

The Geoconnections program is responsible for developing the *Operational Architecture* and the *Technical Architecture*. The *Systems Architecture* design is left to each of the individual participating organizations. The specification of services and interfaces that forms the *Technical Architecture* will allow organizations to develop and/or implement compliant components or systems that offer and/or use these services. The CGDI will be described in the *Technical Architecture* in terms of standards-based services that are well defined and whose behavior and relationships are well understood. Each package represents one or more standardized architectural services or protocols. With the understanding of the underlying IT architecture, GIS components and systems can be designed and implemented to either make use of CGDI services or to extend those services.

4.4 Developing Components, Systems, and Applications

When using the CGDI, clients will use services that obtain and manipulate data, but this will be done using applications that access the physical implementations of these services. These physical implementations are called servers.

Developers will use the service and interface specifications to develop compliant components, and deploy them to offer CGDI-compliant services. For example, a CGDI data provider will provide metadata through a server that complies with the specifications for the Geographic Data Discovery Service. That server could be based on either a software component obtained from another organization, or it could be based on a custom component developed entirely in-house.

Application developers would also use the service and/or interface specifications, in order to develop end-user applications. It will be possible to chain together any number of CGDI services in order to provide a wide variety of specialized systems and applications.

4.5 CGDI Development Network

A CGDI development network exists that provides an environment to test emerging specifications and showcase operational servers. The development network is accessible from the CGDI architecture web site, at http://www.geoconnections.org/architecture/.

5 TECHNICAL ARCHITECTURE

The technical architecture continues to evolve. More than 40 services are under development. Ultimately, each service will have a complete description, and the interfaces will be defined using Web Services Description Language (WSDL). WSDL is further described in Appendix 2.

The development status and latest version of each proposed service can be found on-line, at http://www.geoconnections.org/architecture/

5.1 Endorsed Specifications

The services with complete implementation specifications that have been endorsed by TAP are:

- Web Map Service (WMS)
- Web Feature Service (WFS)
- Geodata Discovery Service
- Geography Markup Language (GML)
- Geodata Resource Registry (GEO metadata profile)

6 OPERATIONAL ARCHITECTURE

As identified in the earlier sections of this document, the CGDI will be implemented as a network of co-operating physical servers, providing services (and data via services) in such a way that:

1) One can build an application that makes use of these services (and thus save development time, development costs, operating costs, and have uniformity of data, etc).
2) One can solve a subset of Geographically related problems in an ad hoc way, by using publicly available user interfaces built on top of the underlying services, that provide discovery of, access to, and general "GIS" functionality on top of publicly available data.

CGDI services are intended to be provided by a broad spectrum of agencies. There is no real “center” to the CGDI apart from the consensus-defined specification of “Master” implementations for core services, and the documentation and specifications that are currently being developed for the Geoconnections web site.

As the operational architecture develops, it will further define:

- Who implements the services defined in the technical architecture,
- How organizations go about implementing services in ways that foster interoperability
- the degree of "looseness" of the infrastructure in terms of hierarchies of servers and registration authorities;
- the bindings/linkages that make domain infrastructure and or systems part of CGDI i.e. is it simply implementation of a specific component/service in the prescribed way?
- the role of common or centralized services such as those currently provided by NRCan;
- who can access common/centralized services, and who pays
- the service level guarantees in terms of reliability and response time.

The Geoconnections program advisory network will be instrumental in assisting the development of the CGDI Operational Architecture

GLOSSARY

The terms and definitions used in this document are described below. Although significant effort has been made to be consistent in the use of terms, web services are a rapidly developing field, and the terminology continues to evolve.

Architecture
The organizational structure and operating environment of the CGDI, including the relationships between its parts, and the principles and guidelines governing their design and evolution over time.

Application
A program that performs a specific function directly for a user.

Component
Software that packages the implementation of a service and can provide the realization of a set of interfaces. A component can include software code (source, binary or executable) or other equivalents such as scripts or command files.

Conceptual Architecture
An overview of the services, data, technology and institutional environment of CGDI. It describes, in general terms, both what the CGDI will include, and how it will operate.

Event
An occurrence of interest to users or developers of the CGDI. Events can be things such as the adjustment of a feature in a framework data layer, a flood in the Red River basin, or the release of a new specification for a CGDI service.

Framework Data
The set of geospatial data that provides the reference framework for all other CGDI compliant geodata. See the CGDI Framework Data Definition.

Geodata
Georeferenced spatial data such as a road network or a satellite image. Geodata explicitly describes the spatial extent of a set of features or describes a measurable surface. It includes both geospatial data and geolinked data.

Geolinked Data
Data that is referenced to an identified set of geographic features without including the spatial description of those features. Geolinked data is normally attribute data in tabular data (such as population counts) that refers to a known framework (such as provinces), where the elements (the provinces) are refered to by their unique identifier (such as the province name). Geolinked data refers to

all attribute data that is not directly attached and bundled with the geographic coordinates to which it applies.

Geospatial Data
Data with explicit geographic positioning information included, such as a road network from a GIS, or a georeferenced satellite image. Geospatial data may include attribute data that describes the features found in the dataset.

Interface
A specification for a set of operations that are made externally available by a component to other components. The state and functionality of a component is hidden, and is only made externally accessible through the interfaces of the components. The interfaces are the only "public" or "visible" part of the component. The same interface may be provided by several components and used by many components or applications.

Reference Architecture
A technical blueprint that identifies and defines the services that comprise the CGDI, and specifies the interfaces to those services.

Registry
A listing of the individual datasets, services, or other things made available by an organization to users of the CGDI. There are two kinds of registries:

> **Type Registry:** A listing of the different types (classes) of objects, such as services, components, or events, which are recognized by CGDI services or applications.
>
> **Instance Registry:** A listing of the individual services, components, datasets, or other things that comprise the CGDI or are relevant to its users. Registries are used to identify, locate, and describe individual instances.

Server
A physical installation of a component that delivers a service, and provides the realization of its operations.

Service
A collection of operations, accessible through one or more interfaces, that allows a user to evoke a behavior of value to that user. A service is delivered by a server.

Site
A location (e.g. URL) at which a system is accessed.

System
Servers and data organized to accomplish one or more services. A system may be accessible at more than one site.

APPENDIX 1: AN ARCHITECTURE BASED ON WEB SERVICES

The CGDI is to be based upon a "web service architecture". This document: defines the needs that led to the development of web service architectures; describes the attributes of web service architectures in general, and the CGDI in particular;

- describes the benefits of this architecture; and
- provides an example showing how to develop an applications using this architecture.

A1.1 The need for "web service architectures"

Web service architectures arose from the need for computer-to-computer communication between distributed organizations. The Internet has vastly increased person-to-person communication. It has also fueled the demand for computer-to-computer communication, but progress in this area has lagged due to the lack of a ubiquitous communication mechanism. The mechanism must accommodate all that wish to participate, no matter what computer platform they use. The barriers to participation must be low. Historical solutions to this need, e.g. CORBA, have had too high an entry barrier.

These needs are paramount for B2B e-commerce, and this market is fueling the development of web service architectures. The same needs are also imperative to the establishment of distributed infrastructures such as the CGDI. The CGDI is purposed to increase the on-line availability of geospatial data and services, and foster new geomatics applications. Open computer access to distributed data and other geomatics services is crucial to meet this goal.

A1.2 Web Service Architecture Attributes

Web service architectures provide a distributed environment in which services can be deployed and invoked using standard Internet protocols. A service is a reusable encapsulations of cohesive functionality with a well-defined programmatic interface.

The key concepts in this definition are:
Internet Protocol: Web services invocation is based on ubiquitous Internet transport mechanisms such as HTTP and XML.
Distribution: Web services can be deployed on any computer that is connected to the Internet, and can potentially be invoked by any other computer connected to the Internet.
Reusability: The services are units of functionality that are potentially reusable in many different applications.
Cohesion: All the functions performed by the service are related to on another, and are not tightly bound to other services.

Programmatic Interface: The service has a well-specified contract that it obeys, and this interface is suitable for programmatic access.

A Web service architecture must fundamentally define a distributed computing platform in which web services can be deployed and invoked. More sophisticated architectures provide additional capabilities such as service discovery, security and quality of service. There are a number of emerging web service architectures such as IBM's Web Service Conceptual Architecture

http://www-4.ibm.com/software/solutions/webservices/pdf/WSCA.pdf

and Microsoft's .net

http://www.microsoft.com/net/default.asp .

A1.3 The CGDI architecture

The CGDI is a web service architecture instance, purposed for the Canadian geospatial domain. The architecture is in the formative stage. The CGDI, as a web service architecture, must provide an environment in which web services can be deployed and invoked. The CGDI could define its own platform, but instead intends to leverage of the efforts of other endeavors (such as B2B e-commerce, and the OGC), and adopt a general-purpose and widely supported platform when one is available. In the absence of a widely supported general-purpose platform, the CGDI today is based on the platform described in the draft OGC Basic Service Model
(http://feature.opengis.org/members/archive/arch01/01-022r1.pdf).
This platform is summarized as follows:

A service is deployed via a URL accessible through a Web server.
Services are invoked via an HTTP "get" or "post".
When using a "get", parameters are passed using the CGI conventions.
When using a post, parameters are XML encoded.
The service must support a GetCapabilities operation that returns service metadata in XML form.

The OGC is evolving this platform and working to align it with other emerging web service architectures. One recent aspect of this alignment has been the adoption of Web Services Description Language (WSDL) for service definition (see Appendix 2).

Standard web service platforms do not, nor will not, provide some capabilities that are crucial to the CGDI. The missing capabilities relate mostly to the fact that, within a geospatial infrastructure, **data** deserves the same prominence as **services**. A general-purpose web service platform will allow **services** to be discovered and invoked, but the CGDI must also allow **data** to be discovered. **Data** will, of course, be discovered through **services**, but these **services** are sufficiently important to be considered part of the CGDI architectural platform (just as service discovery services are part of a standard web services platform). The CGDI web services platform thus provides services to:

- Register geospatial data;
- Discover geospatial data.

The CGDI will also spearhead the identification and specification of web services specific to the geospatial domain.

A1.4 Benefits of a "web service architecture"

Web service architectures leverage off the pervasive Web, providing a ubiquitous distributed computing platform. Applying distributed computing no longer means a heavy financial and training investment in technologies such as CORBA. Many methods exist to publish legacy applications to the Web. A service can be made web accessible no matter how it is implemented, or what platform it executes on. Applications can be easily built from services running on heterogeneous platforms in any location. These benefits are vital for large organizations (corporations or governments) whose distributed divisions, running different computing platforms, need to inter-operate. Inter-organizational infrastructures, such as the CGDI and B2B infrastructures become possible through web service architectures.

Applying web service architectural principals to development yields applications composed of loosely coupled, distributed (or distributable), and reusable services. Complex applications are decomposed into simpler entities that can be independently developed. The development adds to the pool of services that become available for use in new, even more sophisticated applications. The cost of application development is reduced, making sophisticated application development more cost effective.

A1.5 Example of how to exploit the "web service architecture"

This example demonstrates how an application can be built using a "web service architecture". The case study develops a web application that accepts place names, and returns a map showing a road route between the two places. This application might be implemented as shown in
Figure **1**.

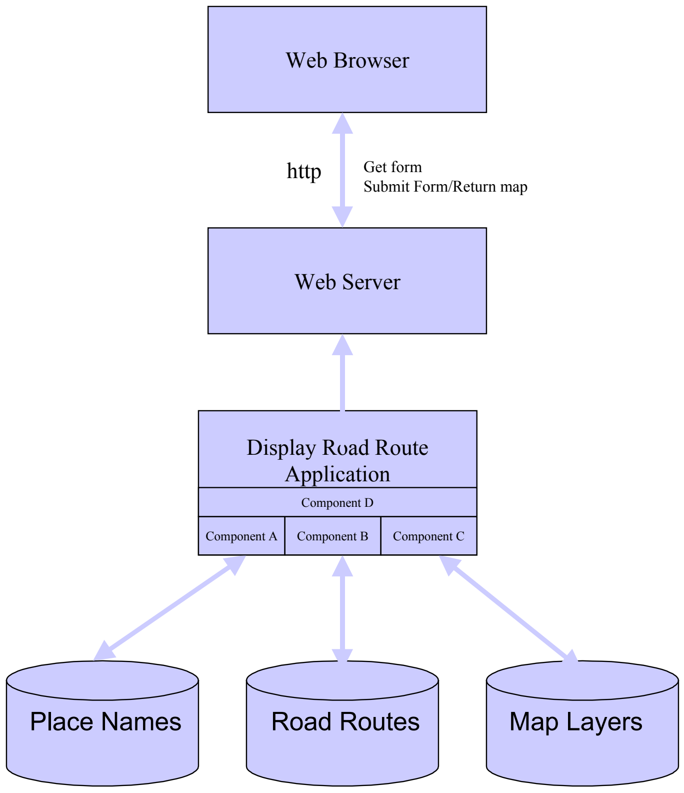

Figure 1: Traditional Application Design

The user invokes the application through a web browser, and enters the place names. The web server invokes an application, which performs the necessary processing, (accessing a database of place names, a road route database, and a database of other map features) and returns a map to the browser as a GIF image. For this discussion, the important aspect is the way in which the application is designed, particularly the partitioning of the software into components and the interfaces between the components. Traditional design techniques would structure the software into logical units with well-defined interfaces, but would deploy the application as a monolithic entity, not suitable for distribution.

Applying web-service based technique to this application, would decompose the application into a number of web service components, which are deployed as standalone servers with an http interface. The result of this architecting is shown in
Figure . The users access to the web browser, and the browser to web server interaction is unchanged, and hence is not shown. The application has been constructed to make use of the services provided by distinct servers:

- A gazetteer server provides services to convert a place name into a geographic location;

- A route server provides services to get the road route between two geographic locations;
- A map server provides services to get a base map on which the route will be displayed;
- An overlay server provides a service to overlay the route on the map;
- A display route server makes use of the above to convert a road route request into a map displaying the route.

The services are deployed in the web service platform and hence can potentially be located anywhere on the Internet, even though for the initial deployment they might all reside on a single host computer.

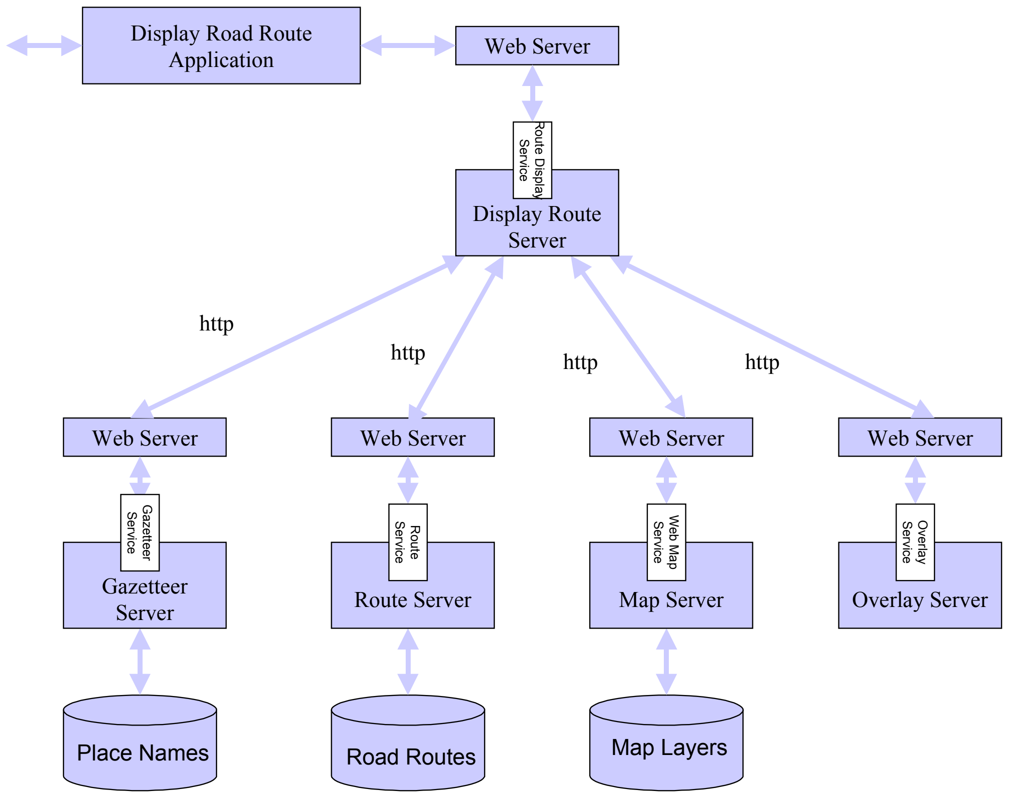

Figure 2: Application Design using Web Services Architecture

The benefits of this architecture are not necessarily apparent when one considers this application in isolation. However, when the application is considered as part of a distributed infrastructure such as the CGDI, it is immediately obvious that this is a better architecture:

- The development of this application has resulted in the deployment of a number of reusable services that can bootstrap the development of other application.

- The architecture is scalable - even if all services were initially deployed on a single computer, and heavy use swamps the server, services can easily be migrated to different computers.
- The back-end services can be provided the most suitable participant (e.g. the Atlas of Canada can provide the server that provides that base map).
- Additional benefits are realized when the service interfaces become "standards" (formal or de-facto), and there are multiple deployments of the same services. The design allows the application to make use of new and better implementation of the services as they emerge.

APPENDIX 2: DESCRIBING WEB SERVICES

Web services are self-describing, modular applications that can be published, located, and invoked across the web [1]. They have been characterized as building blocks for creating open, distributed systems. The Web Services Description Language (WSDL, the successor to NASSL) is basically an interface definition language for network-based services; it specifies *what* type of behaviour a service provides, not *how* it is implemented. Given that the CGDI architecture is likely to be decidedly "service-centric", perhaps we should align our efforts with emerging standards related to service-oriented architectures. I propose we proceed as follows:

- describe non-operational service info using the existing template (service classification, scope of application, provider, etc.)
- describe operational service info using WSDL (described below)

A2.1 Elements of a service-oriented architecture

As platforms become more diverse (i.e. different platforms, devices, connections) technologies like XML and SOAP become crucial: the former for handling structured or semi-structured data, the latter for invoking services in a robust and flexible manner. This reflects the trend toward systems of loosely coupled, dynamically bound components. Other technologies such as UDDI (Universal Discovery Description and Integration—a specification for registries) and WSDL play a role in service discovery. These comprise the basic infrastructure for web services: providers can describe themselves, service requesters can describe what they're looking for, and service brokers can determine how to match them up.

A service-oriented architecture encompasses three main roles: service providers, service brokers (which match requestors with providers, perhaps according to some access control model), and service requestors. The intent is to promote interoperability—the interaction between a service provider and a service requester is designed to be platform and language independent. A WSDL document defines the interface and describes the service, including supported network protocols (usually HTTP). The fundamental operations are then: publish, find, and bind.

6.2 Describing web services with WSDL

The functions provided by a web service can be described using WSDL. Version 1.1 was recently published as a W3C Note [2], and this document will apparently be forwarded to a W3C working group and enter into the formal web standards review process. WSDL is an XML-based format for describing network services. The current specification document describes the core service definition framework plus binding extensions for several protocols and message formats (e.g. SOAP 1.1, HTTP GET/POST, MIME). From the introduction to the spec:

> *As communications protocols and message formats are standardized in the web community, it becomes increasingly possible and important to be able to describe the communications in some structured way. WSDL addresses this need by defining an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication. (sec. 1)*

A WSDL document describes what a web service can do, where it is located, and how to invoke it. Consider a service that provides the exchange rate between any two currencies; such a service is available from the Xmethods web site (http://www.xmethods.net/detail.html?id=5). Here is the WSDL document for the service:

```
<?xml version = "1.0"?>
<definitions name="CurrencyExchangeService"
  targetNamespace="http://www.xmethods.net/sd/CurrencyExchangeService.wsdl"
  xmlns:tns="http://www.xmethods.net/sd/CurrencyExchangeService.wsdl"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <message name="getRateRequest">
    <part name="country1" type="xsd:string"/>
    <part name="country2" type="xsd:string"/>
  </message>
  <message name="getRateResponse">
    <part name="return" type = "xsd:float"/>
  </message>

  <portType name="CurrencyExchangePortType">
   <operation name="getRate">
      <input message="tns:getRateRequest" name="getRate"/>
      <output message="tns:getRateResponse" name="getRateResponse"/>
    </operation>
  </portType>

  <binding name="CurrencyExchangeBinding"
    type="tns:CurrencyExchangePortType">

    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="getRate">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="encoded" namespace="urn:xmethods-CurrencyExchange"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </input>
      <output>
        <soap:body use="encoded" namespace="urn:xmethods-CurrencyExchange"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </output>
    </operation>
  </binding>

```

```
42   <service name="CurrencyExchangeService">
43     <documentation>
44       Returns the exchange rate between the two currencies
45     </documentation>
46     <port name="CurrencyExchangePort" binding="tns:CurrencyExchangeBinding">
47       <soap:address location="http://services.xmethods.net:80/soap"/>
48     </port>
49   </service>
50 </definitions>
```

The <definitions> element is the root element, and it contains three main sections:

<types>: an optional section that defines any complex datatypes the service uses—these are defined using the XML Schema definition language

<message> and <portType>: what operations are provided

<binding>: how the operations are invoked

<service>: where the service is located

A <portType> element corresponds to one or more operations, where an operation is expressed as a specific input/output message sequence. We see that the getRate operation (lines 18-21) accepts a getRateRequest message as input and returns a getRateResponse message as output.

A <binding> element implements a <portType> using a particular protocol (e.g. SOAP, CORBA); the type attribute (line 25) must match the name of a defined <portType> (line 17). This binding uses SOAP. A service is modeled as a collection of ports.

NOTE: toolkits can be used to create client stubs that simplify access to a service, or generate WSDL files from a component.

References

[1] *Web Services Architecture Overview* (IBM, September 2000

http://www-106.ibm.com/developerworks/webservices/library/w-ovr/?dwzone=webservices

[2] *Web Services Description Language (WSDL) 1.1*. W3C Note (15 March 2001). http://www.w3.org/TR/wsdl.