Prepared by

galdos systems inc

Transportation Spatial Information Infrastructure

TP 14388E

Prepared for

Transport Canada

April 2005

David S. Burggraf, Ph.D.
Suite 1300, 409 Granville Street
Vancouver, BC V6C 1T2
Canada

Transportation Spatial Information Infrastructure

TP 14388E

April 2005

Notices

This project is part of Canada's Intelligent Transportation Systems (ITS) R&D Plan, Innovation Through Partnership, funded by the ITS Office of Transport Canada under the Strategic Highway Infrastructure Program (SHIP).

The Transportation Development Centre of Transport Canada served as technical authority for this project.

Project Team

Galdos Systems Inc.

David S. Burggraf, Ph.D., Director, Research and Development

Terence Guo, Software Engineer

Wes Kubo, Software Engineer

University of Toronto

Baher Abdulhai, Ph.D., Associate Professor, Director, Toronto Intelligent Transportation Systems (ITS) Centre

Simon Foo, M.Sc. Student

BC Transit

Clark C. Lim, P.Eng., Senior Transportation Engineer

InCom Korea

Byeong Yong Rhee, CTO, InCOM Korea

Korea Highway Corporation

Sukkee Hong, M.Sc., Tranportation Engineer

Un sommaire français se trouve avant la table des matières

Transport Canada Transports Canada

PUBLICATION DATA FORM

1. Transport Canada Publication No. TP 14388E	2. Project No.	3. Recipient's Catalogue No.
4. Title and Subtitle Transportation Spatial Information Infrastructure		5. Publication Date April 2005
		6. Performing Organization Document No.
7. Author(s) David S. Burggraf		8. Transport Canada File No. 2450-GP011
9. Performing Organization Name and Address Galdos Systems Inc. Suite 1300, 409 Granville Street Vancouver, British Columbia Canada V6C 1T2		10. PWGSC File No. 052ss-T8663-030015
		11. PWGSC or Transport Canada Contract No. T8663/030015/004/SS
12. Sponsoring Agency Name and Address ITS Office – Transport Canada Place de Ville, Tower C, Floor 27 330 Sparks Street Ottawa, Ontario Canada K1A 0N5		13. Type of Publication and Period Covered Final
		14. Project Officer Pierre Bolduc

15. Supplementary Notes (Funding programs, titles of related publications, etc.)

This project is part of Canada's Intelligent Transportation Systems (ITS) R&D Plan, *Innovation Through Partnership*, funded by the ITS Office of Transport Canada under the Strategic Highway Infrastructure Program (SHIP). The Transportation Development Centre of Transport Canada served as technical authority for this project.

16. Abstract

A prototype Transportation Spatial Information Infrustructure (TSII) was developed and implemented in this R&D project, based on XML and the emerging standards from the Open Geospatial Consortium (OGC). The components of this TSII include multiple OGC Web Services: two Web Map Services (WMS), a Web Registry Service (WRS), and four Web Feature Services (WFS), which use a standard data transfer format, Geography Markup Language (GML), for the encoding of geographic information. The use of GML as the transfer format enabled the integration of transportation data from multiple sources with different underlying data storage formats, which was then delivered as distinct features and displayed in real time in a standard Web browser. The TSII prototype employs a WMS to deliver real-time traffic data as a Scalable Vector Graphics (SVG) map in a Web browser. Refreshing the SVG map pulls current data from the Ontario Ministry of Transportation (MTO) and City of Toronto traffic sensors from a WFS and redisplays the map's colour-coded traffic features. Interoperability is one of the main advantages of using of an open standard solution and is demonstrated in this project by integrating data from various authorities such as MTO, City of Toronto, and Intergraph.

17. Key Words Geography Markup Language, GML, Web Feature Service, WFS, Web Map Service, WMS, Intelligent Transportation Systems, ITS, Open Geospatial Consortium, OGC, spatial information infrastructure, sensor data, Geographic Information Systems, GIS	18. Distribution Statement Limited number of copies available from the Transportation Development Centre Electronic version available from Transport Canada Web site: www.tc.gc.ca

19. Security Classification (of this publication) Unclassified	20. Security Classification (of this page) Unclassified	21. Declassification (date) —	22. No. of Pages xxxiv, 38	23. Price Shipping/ Handling

Transports Canada Transport Canada

FORMULE DE DONNÉES POUR PUBLICATION

1. N° de la publication de Transports Canada TP 14388E	2. N° de l'étude	3. N° de catalogue du destinataire
4. Titre et sous-titre Transportation Spatial Information Infrastructure		5. Date de la publication Avril 2005
		6. N° de document de l'organisme exécutant
7. Auteur(s) David S. Burggraf		8. N° de dossier - Transports Canada 2450-GP011
9. Nom et adresse de l'organisme exécutant Galdos Systems Inc. Suite 1300, 409 Granville Street Vancouver, British Columbia Canada V6C 1T2		10. N° de dossier - TPSGC 052ss-T8663-030015
		11. N° de contrat - TPSGC ou Transports Canada T8663-030015/004/SS
12. Nom et adresse de l'organisme parrain Bureau STI – Transports Canada Place de Ville, Tour C, 27e étage 330, rue Sparks Ottawa, Ontario Canada K1A 0N5		13. Genre de publication et période visée Final
		14. Agent de projet Pierre Bolduc

15. Remarques additionnelles (programmes de financement, titres de publications connexes, etc.)

Ce projet fait partie du Plan de R&D du Canada sur les systèmes de transports intelligents (STI), *Innover par l'établissement de partenariats*, financé par le Bureau des STI ainsi que par Transports Canada dans le cadre du programme stratégique d'infrastructures routières (PSIR). Le Centre de développement des transports de Transports Canada a agi comme responsable technique pour ce projet.

16. Résumé

Ce projet de R&D a consisté à développer et mettre en oeuvre un prototype d'infrastructure de données spatiales appliquée aux transports (IDST), fondée sur le langage XML et sur les nouvelles normes de l'OGC (*Open Geospatial Consortium*). Cette IDST fait appel à plusieurs services Web de l'OGC : deux services de cartes Web (WMS, pour *Web Map Services*), un service de registres Web (WRS, pour *Web Registry Service*) et quatre services d'entités Web (WFS, pour *Web Feature Service*), qui utilisent un format standard de transfert des données, et le langage de balisage géographique (GML, pour *Geography Markup Language*) pour coder les données géographiques. L'utilisation du GML comme format de transfert a permis l'intégration de données provenant de plusieurs sources et stockées sous des formats différents, la transmission de celles-ci en tant qu'entités distinctes et leur affichage en temps réel sur un fureteur Web standard. L'IDST a recours à un WMS pour transmettre des données de circulation en temps réel sous forme de graphiques vectoriels adaptables ou cartes SVG (*Scalable Vector Graphics*) sur un fureteur Web. Le fait de rafraîchir la carte SVG extrait d'un WFS les données courantes provenant des détecteurs de circulation du ministère des Transports de l'Ontario (MTO) et de la Ville de Toronto et affiche les entités de circulation à jour par des codes couleur sur la carte. L'interopérabilité s'avère l'un des principaux avantages d'une solution à norme ouverte. Ce projet, qui intègre les données de diverses administrations, telles le MTO, la Ville de Toronto et Intergraph, en est une démonstration éloquente.

17. Mots clés Langage de balisage géographique, GML, service d'entités Web, WFS, service de cartes Web, WMS, systèmes de transports intelligents, STI, *Open Geospatial Consortium*, OGC, infrastructure de données spatiales, données de capteurs, systèmes d'information géographique, SIG.	18. Diffusion Le Centre de développement des transports dispose d'un nombre limité d'exemplaires. Version électronique disponible à partir du site Web de Transports Canada : www.tc.gc.ca

19. Classification de sécurité (de cette publication) Non classifiée	20. Classification de sécurité (de cette page) Non classifiée	21. Déclassification (date) —	22. Nombre de pages xxxiv, 38	23. Prix Port et manutention

CDT/TDC 79-005
Rev. 96

Canadä

Acknowledgements

The support and advice from Pierre Bolduc, Transportation Development Centre of Transport Canada, is gratefully acknowledged. The cooperation from Intergraph for permitting access to its WMS in this project is also greatly appreciated.

Executive Summary

Introduction

The focus of this R&D project was to develop an ITS spatial infrastructure, based on emerging spatial standards from the Open Geospatial Consortium (OGC) and the World Wide Web Consortium, in a manner consistent with the Canadian ITS Architecture. The OGC Web services used in this R&D project use a standard data transfer format, Geography Markup Language (GML), soon to be an ISO standard (ISO/TC 211 19136) for the encoding of geographic information.

GML is an eXtensible Markup Language (XML) application that is fast becoming the world standard for geographic information delivery over the Internet. Using GML as a data transfer format, geographic information can be integrated from multiple application domains, regardless of what the underlying data storage formats are. This integrated data can then be delivered via web service transactions as distinct features and displayed in a Web browser using any selected map style.

The main goal of this project was to develop a prototype Transportation Spatial Information Infrustructure (TSII) that employs two OGC Web Map Services (WMS) to deliver real-time traffic data as a Scalable Vector Graphics (SVG) map to the end user. By refreshing the SVG map, updated readings from traffic sensors are pulled from a Web Feature Service (WFS) and redisplayed on a colour-coded traffic congestion map. A Web Registry Service (WRS) is an online web service used to discover the component WFSs, WMSs and GML data sets of the TSII. More specifically, the TSII comprises the following:

- A GML Application Schema for ITS based on the Canada ITS Data Element Registry, representing a simplified traffic model chosen to illustrate the handling and integration of dynamic traffic information;
- OGC-compliant implementations of a WFS to integrate and distribute geographic information encoded as GML;
- OGC-compliant implementations of a WMS used to style and display GML data received from a WFS;
- An OGC-compliant implementation of the ebRIM profile of the Catalogue for the Web, otherwise known as a WRS used to discover Web services and GML datasets related to the TSII;
- A WFS Gateway, which converts real-time Ontario Ministry of Transportataion (MTO) and City of Toronto sensor data to GML and transfers this to a WFS.

One of the main advantages of adopting an open standard solution over a proprietary one is interoperability, which leads to lower overall costs of maintaining data. Interoperability is demonstrated in this project by integrating static and dynamic transportation data from multiple sources. The sources of data in this project are as follows:

- Road and highway data, including geometry and nonspatial metadata obtained from NAVTEQ, a well-known data supplier worldwide;
- Road and waterbody map data from an ArcSDE/Oracle database delivered via an OGC-compliant WMS hosted by the GIS vendor, Intergraph;
- Loop detector data from MTO, including identification, functionality, real-time sensor readings, and location along Highway 401 in the metro Toronto area;
- Changeable Message sign information from MTO, including real-time message display, identification, and location along Highway 401 in the metro Toronto area;
- Loop detector data from the City of Toronto, including identification, functionality, real-time sensor readings, and location along Don Valley Parkway, Gardener Expressway, and Lake Shore Drive;
- Changeable Message sign information from the City of Toronto, including real-time message display, identification, and location along Don Valley Parkway, Gardener Expressway, and Lake Shore Drive;
- Traffic incident reports from MTO;
- Traffic incident reports from the City of Toronto.

Note that the TSII enables other sources of static and dynamic data (weather sensors, GPS devices, etc.) to be integrated with the existing traffic data, which could then be displayed with the traffic information on the same map.

TSII Components and Architecture

The TSII prototype is implemented using four Web Feature Servers, two Web Map Servers and one Web Registry Server at Galdos Systems, and the WFS Gateway at the University of Toronto ITS Lab. Data and communication between these servers in this model are intended to represent the dynamic and distributed data handling of a real-world Intelligent Transportation System.

Figure 1 gives a high-level overview of the TSII prototype architecture. The coloured boxes represent system servers and the labelled arrows indicate the type of transaction executed between the servers. The description of each server/system is summarized in Table 1.

Table 1 Description of System Servers

System/Server	Description
WFS Gateway	Converts traffic sensor, incident and CMS data to WFS transactions (Insert, Update) to the dynamic traffic model and incident WFSs
Static TSII Object DB	A WFS that stores and distributes static objects such as sensors and road network data representing the 400 series highways
Traffic Model DB	A WFS that stores and distributes traffic link data
Observation DB	A WFS that stores and distributes loop detector observations, incident reports and current CMS displays
CRS Registry	A WRS that stores and allows for the discovery of CRS data (2D, and 1D Linear Reference Systems) and transformations
FreeStyler WMS	A WMS that styles and displays GML data as a map
Listener	A component of a WFS that detects an incoming transaction and triggers an appropriate action (e.g. replicate or create new feature)

The description of each WFS transaction is summarized in Table 2.

Table 2 Description of WFS Transactions

Transaction	Description
Insert Feature	Adds a new feature or features with unique ID to the WFS
Update Feature	Modifies an existing feature or features in the WFS
Get Feature	Retrieves a feature or features from the WFS. The features retrieved can be selected by feature type, feature ID, spatial location, or by arbitrary property/attribute values.
Describe Feature	Retrieves the GML schema definition of the feature or features from the WFS
Delete Feature	Deletes an existing feature or features in the WFS

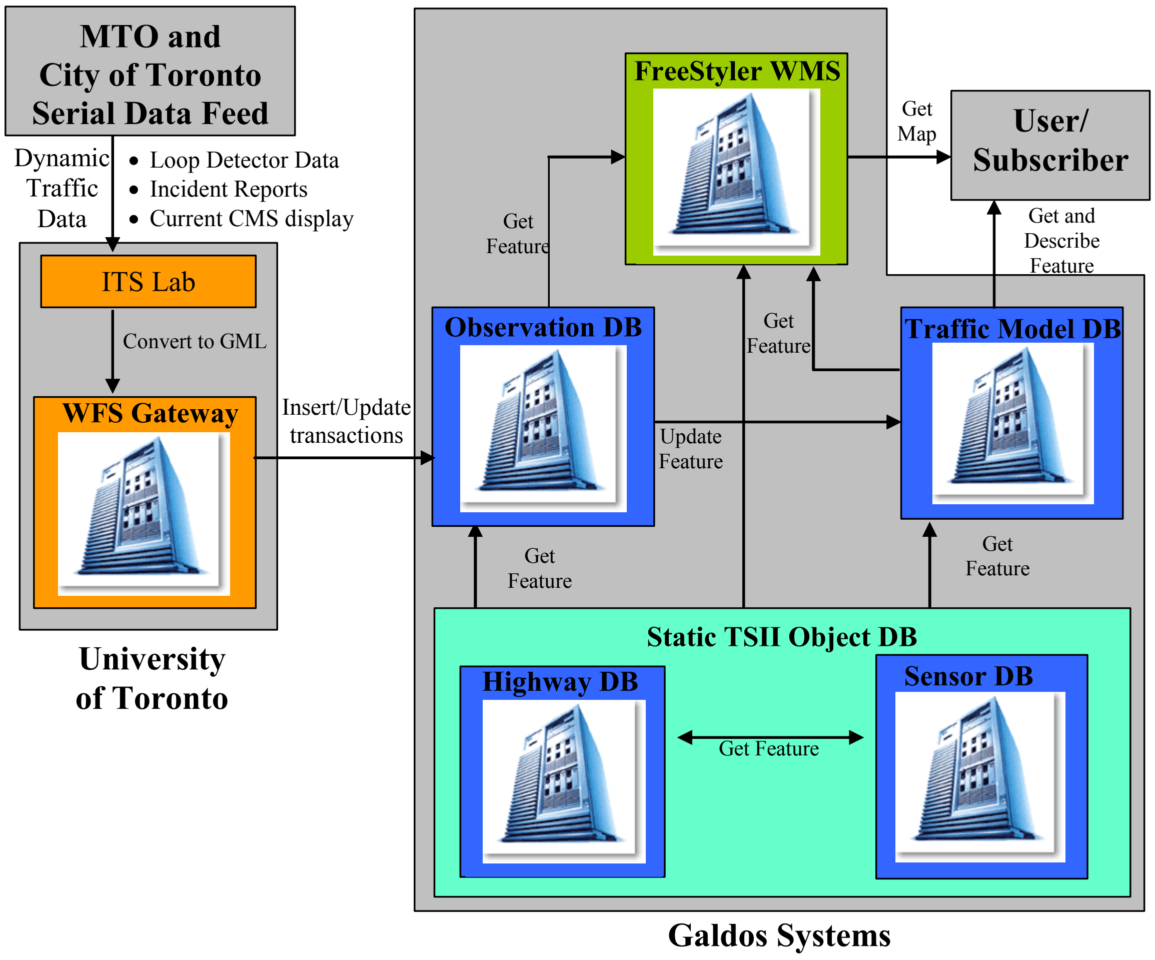


Figure 1 Overview of TSII Prototype Architecture

The Web Feature Service Gateway resides at the University of Toronto and consists of PHP scripts developed by the ITS Lab at the University of Toronto to convert the real-time MTO and City of Toronto data to GML Observations. Insert transaction requests are then created with the newly created XML file as payload and wrapped in the appropriate `<wfs:transaction>` tags. Java classes developed by Galdos Systems Inc. automatically submit the WFS Insert transaction over the Internet using the http post protocol.

The traffic data is delivered to the ITS Lab at the University of Toronto from two serial ports at 19.2k baud, one for MTO and one for the City of Toronto. The physical interface is serial (RS-232)—a multi-kilometre "virtual" serial connection is provided by Siemens OTN fibre optic equipment. The incoming data format is binary, defined by some C structs, which is then serialized. An application then reads the data coming into the ITS Lab and stores it in a local MySQL database.

The main source of dynamic traffic data received by the prototype TSII from MTO and the City of Toronto are loop detector readings. The loop detectors are located in each traffic lane roughly every half kilometre along the Toronto highways. The readings for all loop detectors are taken every 20 seconds, measuring traffic volume, vehicle occupancy and, for most loop detectors, average speed.

The loop detector readings get converted to GML, which are encoded as a special type of Feature called a *LoopDetectorObservation*. The property values of the *LoopDetectorObservation* store the speed, traffic volume and sensor occupancy. Any other transportation features in the TSII that depend on these readings (e.g. the *Link* feature) are automatically created or updated as each *LoopDetectorObservation* is received by the Observation WFS. The *Link* feature represents a traffic link, which is a segment of the highway associated with a cluster of sensors (loop detectors) that cover all lanes in a single direction of traffic. Traffic links store the length of the highway segment, as well as the average speed and transit time along it, where average speed and transit time are derived from the loop detector observations. A “Listener” component of the Observation WFS monitors the transaction log and performs an Insert *Link* feature transaction to the Traffic Model WFS when a *LoopDetectorObservation* is inserted in the Observation WFS.

Incident reports are logged by human operators and conform to a specific file format. The City of Toronto and MTO each have different incident report formats and both are supported by the GML application schema for ITS.

The MTO and City of Toronto provide the current message display of 40 Changeable Message Signs (CMS) at various locations along the Toronto highways. When a CMS display is received by the ITS Lab, the WFS Gateway converts it to a *MessageSignDisplay* feature encoded as GML and submits an Insert/Update WFS transaction request to the Traffic Observation database (WFS) at Galdos Systems. Users of the TSII with a WFS can receive the current *MessageSignDisplay* by retrieving it from the Observation database (WFS) or also by push replication as in the case of incident reports.

The TSII application schema consists of four XSD files organized shown in Figure 2, where solid arrows indicate inclusion of the target schema by the source and dashed arrows indicate import of the target schema by the source.

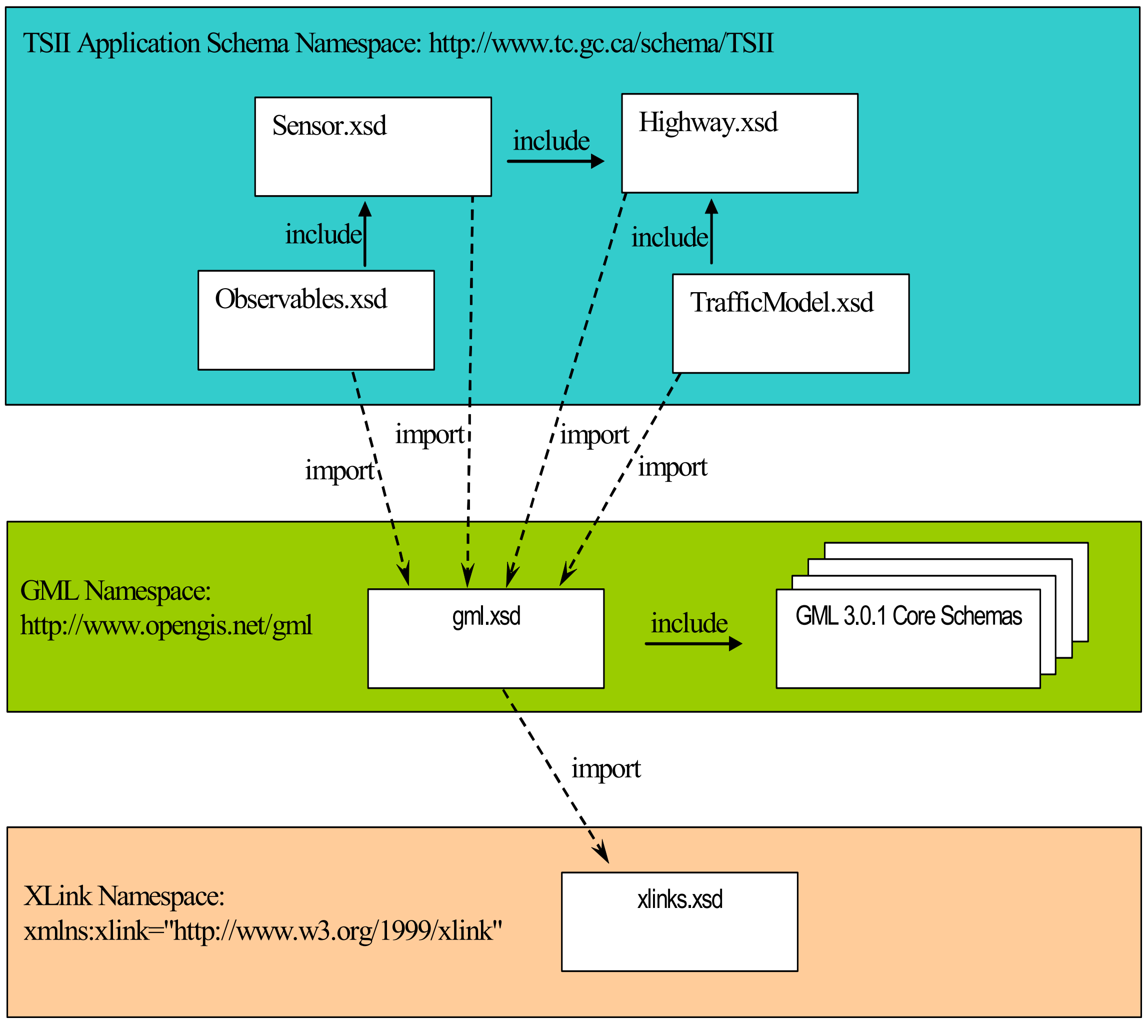


Figure 2 TSII Schema Organization

Each of the four schemas is stored in the WRS and in each of the corresponding Web Feature Servers: Traffic Observation WFS, Traffic Model WFS, Highway WFS, Sensor WFS.

RESULTS

This project combined static road data and real-time loop detector readings in a simple traffic scenario and implemented many of the basic components of the TSII using the four WFSs, two WMSs and the WFS Gateway. WFS software extensions were required and implemented to properly support the automatic update of dynamic features. In particular, the Listener/Action Manager component of Cartalinea (the Galdos WFS implementation) was developed in this project. The WFS Gateway produces *LoopDetectorObservations* encoded as GML from real-time MTO and City of Toronto sensor readings and supports *IncidentReports* and current *MessageSignDisplays*. The Galdos WMS client displays a road map requested from another OGC-compliant WMS from Intergraph and simultaneously portrays the TSII features including colour-coded traffic congestion overlaid on the road map in a standard web browser.

The WMS client screenshot in Figure 3 portrays the TSII features as SVG overlaid on a road map from an Intergraph WMS to produce a simple traffic flow diagram. When a mouse-over event is detected on a TSII feature, an HTML Tooltips text box is created to expose certain GML properties of the feature using JavaScript. The TSII *Road* features expose only the name (*gml:name* property value) when moused over, as shown in Figure 3.

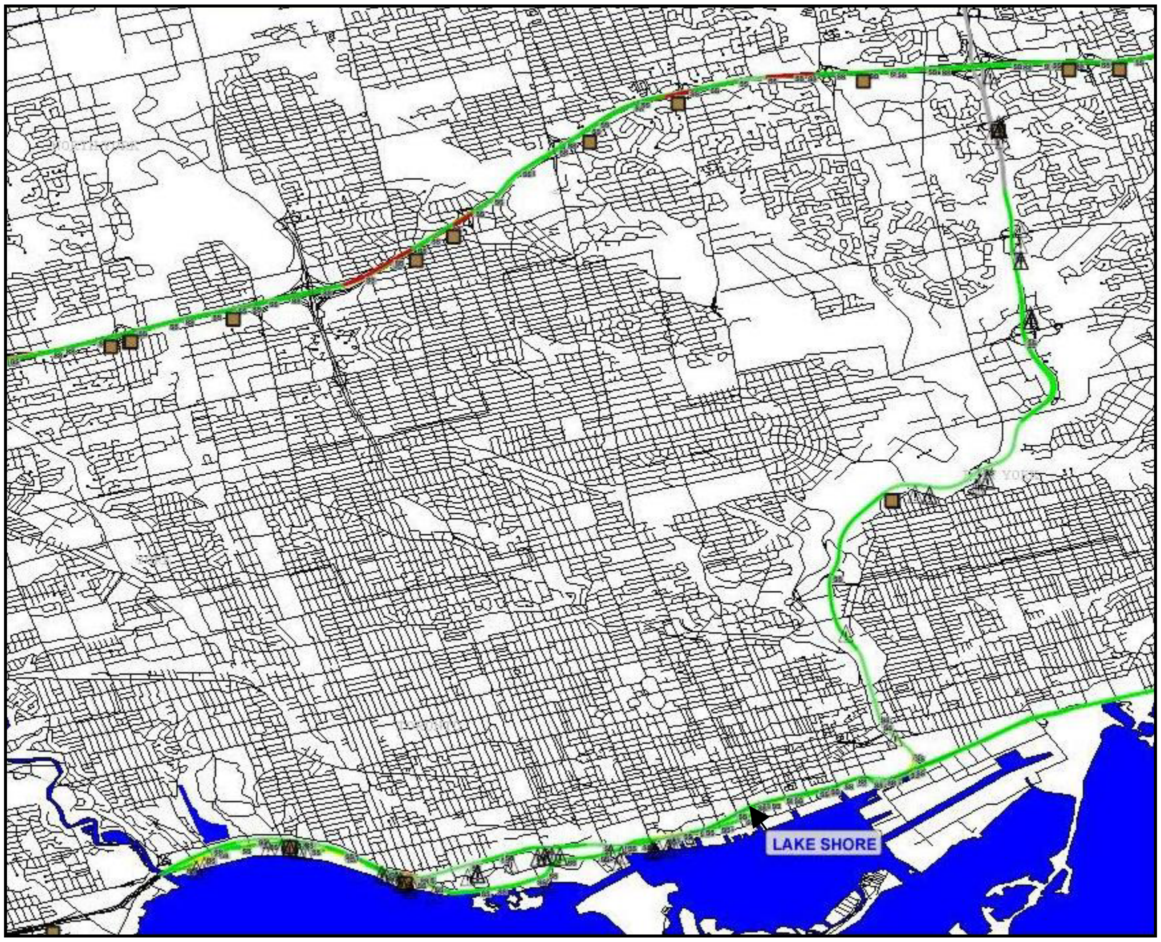


Figure 3 TSII Feature Portrayal in FreeStyler WMS

The traffic *Link* features are segments of the highway that are colour-coded from green to red according to Table 3. The colour-coding scheme of the *Link* features was adapted from that used by the MTO COMPASS Traffic Management System, where the column order is listed in order of priority. If the Average Speed is greater than zero, the colour indicated by the shown cell is used to style the *Link* symbol. If the average speed is zero, then the Traffic Volume column is used to determine the colour. If the Traffic Volume is zero, then the Sensor Occupancy (percentage of the 20 second interval the loop detector is triggered) column is used to determine the colour. In the case that no reading is reported from the sensor in the 20 second interval, the default value "-1" is set for Average Speed, Traffic Volume and Sensor Occupancy, and the corresponding colour code is grey.

Table 3 Colour coding scheme for *Link* features

Average Speed (km/h)	Traffic Volume (vehicle count/20sec/lane)	Traffic Occupancy (%)
> 100	> 0 and < 4	0 - 6
75 - 100	4 - 8	6 - 14
50 - 75	8 - 12	14 - 22
25 - 50	12 - 16	22 - 32
> 0 and < 25	> 16	> 32
-1	-1	-1

When moused over, the Tooltips text box lists the values of the following properties and attributes of the *Link* feature:

- gml:id (unique identifier)
- timeStamp (XML Schema dateTime value)
- averageSpeed (in km/h)
- occupancy (%)
- trafficVolume (vehicle count/20 sec/lane)
- averageTransitTime (average time in seconds expected to traverse the *Link*)

ANALYSIS

The overall performance of the TSII prototype can be measured by the total delay from the time of the sensor reading to the instant the data is displayed on screen. Note that each server used in the TSII prototype is a Windows 2000 server that uses dual 2.8 GHz Xeon processors with 2GB DDR SDRAM. The complete data path is illustrated in Figure 4 and is decomposed as a sequence of seven events that are labeled on the diagram.
The average duration for each event is measured and briefly described in Table 4.

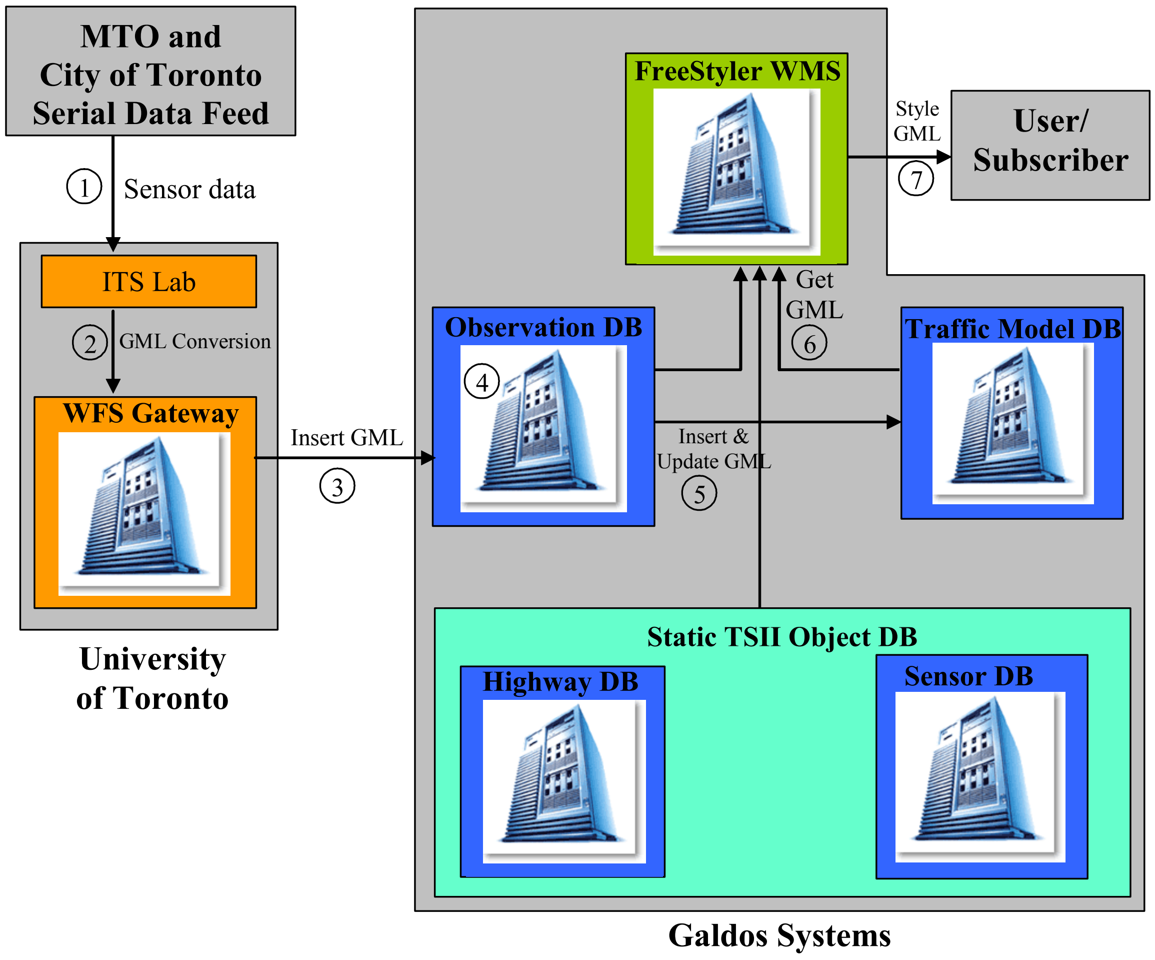


Figure 4 Data Path Decomposition

The first measured event is sensor data preparation and delivery to the ITS lab, which is broken down for the case of loop detectors as follows:

- A vehicle passes over a loop detector embedded under the road surface, creating a disturbance in electric current that is detected by the roadside Advanced Traffic Controller (ATC) device.
- Each ATC collects data (volume, occupancy and speed) over a 20-second interval for several clusters of loop detectors called Vehicle Detector Stations (VDS) in the area.
- A VAX server at MTO receives and stores the data from all ATCs on the highway.
- A process residing on the VAX server polls the database every second and listens for changes. Once a database update is detected, the current data is read and compressed (takes 2.5 seconds on average).
- The compressed data (10 KB on average) is sent through a 19.2 kbps serial link to the University of Toronto ITS lab (transit time is 11 seconds on average).

- The data received at the ITS lab is decompressed and then averaged over all lanes for each VDS before being inserted in a MySQL database (takes 0.9 seconds on average).
- A separate process polls the MTO database every 5 seconds, checking for updated CMS and Incident info. When an update is detected, the data is sent to the ITS lab immediately.

The total time delay corresponding to the sensor data preparation and delivery event is summarized in the first row of Table 4. The corresponding time durations of the remaining six events in the data path are listed in Table 4. Where applicable, the primary protocol or processing technology used in each event is given in parentheses, e.g. (HTTP POST).

Table 4 TSII Prototype Performance Results

EVENT	DESCRIPTION	AVERAGE DURATION (seconds)	
		MTO data	City of Toronto data
1	Sensor data preparation and delivery to ITS lab database	20.0	18.00
2	GML conversion (PHP script)	0.1	0.09
3	Send request via WFS Gateway (HTTP POST) including network delay	1.6	1.10
4	Observation DB commitment (XHIVE Database)	171.0	110.50
5	Traffic DB update (Java based listener/action manager). This event is concurrent with event 4.	332.5	209.00
6	Get Feature request (HTTP GET)	5.5	5.00
7	Transform GML to SVG (XSLT)	2.1	2.00
	Total Time	**361.8**	**235.19**

CONCLUSION

The development of the prototype TSII based on GML and OGC web services shows that an open standard solution is capable of supporting dynamic traffic sensor information. Such an open standard solution has clear advantages, such as promoting interoperability, but the performance, although not unacceptable, could be improved. The performance results indicate an average lag time from sensor reading to display of roughly 6 minutes for MTO data and 4 minutes for City of Toronto data. There was insufficient time in this project for further investigation into reducing the lag time by optimizing software configuration, for example, which might have yielded better performance results.

The WMS client used in the TSII prototype retrieved and displayed SVG map data from two different web map servers, where the underlying feature data originated in different databases with different storage formats. The WMS hosted by Galdos requested GML feature data from four WFSs with underlying XHIVE (native XML) databases, and the WMS hosted by Intergraph retrieved data from an ArcSDE/Oracle database. The TSII implementation thus demonstrated the seamless integration, distribution, and display of transportation data from multiple sources with different underlying data storage formats, thus enabling wide area plug and play of transportation information. Other sources of static and dynamic data (weather sensors, GPS devices, etc.) in any existing database can be connected to the TSII via WFS and the resulting larger pool of transportation data can then be displayed in the same map or retrieved in the same result set for processing. The OGC open standards-based TSII thus enables wide area plug and play for transportation information.

Sommaire

Introduction

Ce projet de R&D visait à développer une infrastructure spatiale STI (système de transport intelligent) fondée sur les nouvelles normes spatiales de l'OGC (*Open Geospatial Consortium*) et du consortium World Wide Web, et conforme à l'architecture STI pour le Canada. Les services Web de l'OGC utilisés aux fins du projet appliquent un format standard de transfert de données, soit le langage de balisage géographique (GML, pour *Geography Markup Language*), pour le codage des données géographiques. Ce langage est d'ailleurs sur le point de devenir une norme ISO (ISO/TC 211 19136).

Le GML est une application du langage de balisage extensible (XML, pour *eXtensible Markup Language*) qui est rapidement en train de devenir la norme mondiale pour la transmission de données géographiques sur Internet. En utilisant le GML comme format de transfert de données, les données géographiques de plusieurs domaines d'application peuvent être intégrées, quels que soient les formats sous-jacents de stockage des données. Ces données intégrées peuvent ensuite être transmises en tant qu'entités distinctes par des transactions entre services Web, et être affichées par un fureteur Web sur n'importe quel style de carte choisi.

L'objectif principal de ce projet était de développer un prototype d'infrastructure de données spatiales appliquée aux transports (IDST) utilisant deux services de cartes Web (WMS, pour *Web Map Services*) de l'OGC pour fournir à l'utilisateur final des données en temps réel sur la circulation, sous la forme de graphiques vectoriels adaptables ou cartes SVG (*Scalable Vector Graphics*). En rafraîchissant la carte de SVG, on extrait des lectures à jour des capteurs de circulation d'un service d'entités Web (WFS, pour *Web Feature Service*) et celles-ci sont affichées sur une carte des congestions routières à codes couleur. Un service de registres Web (WRS, pour *Web Registry Service*) est un service Web en ligne utilisé pour découvrir les WFS, les WMS et les ensembles de données GML constitutifs de l'IDST. De façon plus précise, l'IDST comporte ce qui suit :

- un schéma d'application GML pour les STI, fondé sur le registre canadien des éléments de données STI, représentant un modèle simplifié de circulation choisi pour illustrer la manipulation et l'intégration de l'information dynamique sur la circulation;
- des mises en œuvre conformes aux normes de l'OGC d'un WFS pour l'intégration et la distribution des données géographiques codées en GML;
- des mises en œuvre conformes aux normes de l'OGC d'un WMS utilisé pour formater et afficher des données GML reçues d'un WFS;
- une mise en œuvre conforme aux normes de l'OGC du profil ebRIM du catalogue pour le Web, aussi connu comme un WRS utilisé pour découvrir les services Web et les ensembles de données GML ayant trait à l'IDST;
- une passerelle WFS qui convertit en GML les données en temps réel des capteurs du ministère des Transports de l'Ontario (MTO) et de la Ville de Toronto, et transfère le tout à un WFS.

Un des principaux avantages d'une solution à norme ouverte par rapport à une solution propriétaire réside dans l'interopérabilité, qui permet de réduire l'ensemble des coûts associés à la mise à jour des données. L'interopérabilité est démontrée dans ce projet par l'intégration de données statiques et dynamiques provenant de multiples sources. Voici les sources sollicitées pour le présent projet :

- données sur les routes et les voies rapides, y compris les métadonnées géométriques et non spatiales obtenues de NAVTEQ, un fournisseur de données reconnu à l'échelle mondiale;
- données de cartes routières et de plans d'eau provenant d'une base de données ArcSDE/Oracle, transmises via un WMS conforme aux normes de l'OGC hébergé chez Intergraph, le fournisseur de SIG;
- données de détecteurs à boucle provenant du MTO : identification, fonctionnalité, lectures de capteurs en temps réel et emplacement le long de l'autoroute 401 dans la région métropolitaine de Toronto;
- information sur les panneaux à message variable provenant du MTO : affichage courant, identification et emplacement le long de l'autoroute 401 dans la région métropolitaine de Toronto;
- données de détecteurs à boucle provenant de la Ville de Toronto : identification, fonctionnalité, lectures en temps réel de capteurs et emplacement le long du Don Valley Parkway, du Gardener Expressway et de Lake Shore Drive;
- information sur les panneaux à message variable provenant de la Ville de Toronto : affichage courant, identification et emplacement le long du Don Valley Parkway, du Gardener Expressway et de Lake Shore Drive;
- rapports d'incidents de la circulation du MTO;
- rapports d'incidents de la circulation de la Ville de Toronto.

Il convient de noter que l'IDST permet l'intégration de données statiques et dynamiques d'autres sources (capteurs météorologiques, appareils GPS, etc.) aux données existantes sur la circulation, lesquelles peuvent alors être affichées sur la même carte que les données de circulation.

Composantes et architecture de l'IDST

Le prototype d'IDST fait appel à quatre services d'entités Web (WFS), à deux serveurs de cartes Web, à un serveur de registres Web chez Galdos Systems, et à la passerelle WFS du laboratoire STI de l'Université de Toronto. Les données de ces serveurs et leur communication au sein du modèle visent à représenter le traitement de données dynamiques et réparties par un système de transport intelligent réel.

La figure 1 donne un aperçu global de l'architecture du prototype d'IDST. Les boîtes de couleurs représentent les serveurs du système et les flèches indiquent les types de transactions exécutées entre serveurs. Le tableau 1 donne une courte description de chacun des serveurs et des systèmes.

Tableau 1 - Description des serveurs du système

Système/Serveur	Description
Passerelle WFS	Convertit les données des capteurs de circulation, des rapports d'incidents et des PMV (panneaux à message variable) en transactions (insertion, mise à jour) visant les WFS du modèle de circulation dynamique et des incidents
Base de données d'objets statiques	WFS qui entrepose et distribue les objets statiques comme les données provenant des capteurs et du réseau routier représentant la série des autoroutes 400
Base de données du modèle de circulation	WFS qui entrepose et distribue les données sur l'état de la circulation
Base de données des observations	WFS qui entrepose et distribue les observations des détecteurs en boucle, les rapports d'incidents et les affichages PMV courants
Registre de coordonnées spatiales	WRS qui entrepose et permet de découvrir des coordonnées spatiales (systèmes de référence linéaire 2D et 1D) et leurs transformations
WMS FreeStyler	WMS qui formate et affiche des données GML sous forme de carte
Auditeur	Composante d'un WFS qui détecte une transaction entrante et déclenche une mesure appropriée (p. ex., reproduit ou crée une nouvelle entité)

Le tableau 2 décrit brièvement chaque transaction WFS.

Tableau 2 - Description des transactions WFS

Transaction	Description
Insertion	Ajoute au WFS une ou des entités à identification unique
Mise à jour	Modifie une entité existante ou des entités existantes du WFS
Lecture	Extrait une entité ou des entités du WFS. Les entités extraites peuvent être choisies par type, par identification, par emplacement spatial ou par des valeurs d'attributs arbitraires.
Description	Extrait la définition du schéma GML d'une entité ou d'entités du WFS
Suppression	Supprime une entité existante ou des entités existantes du WFS

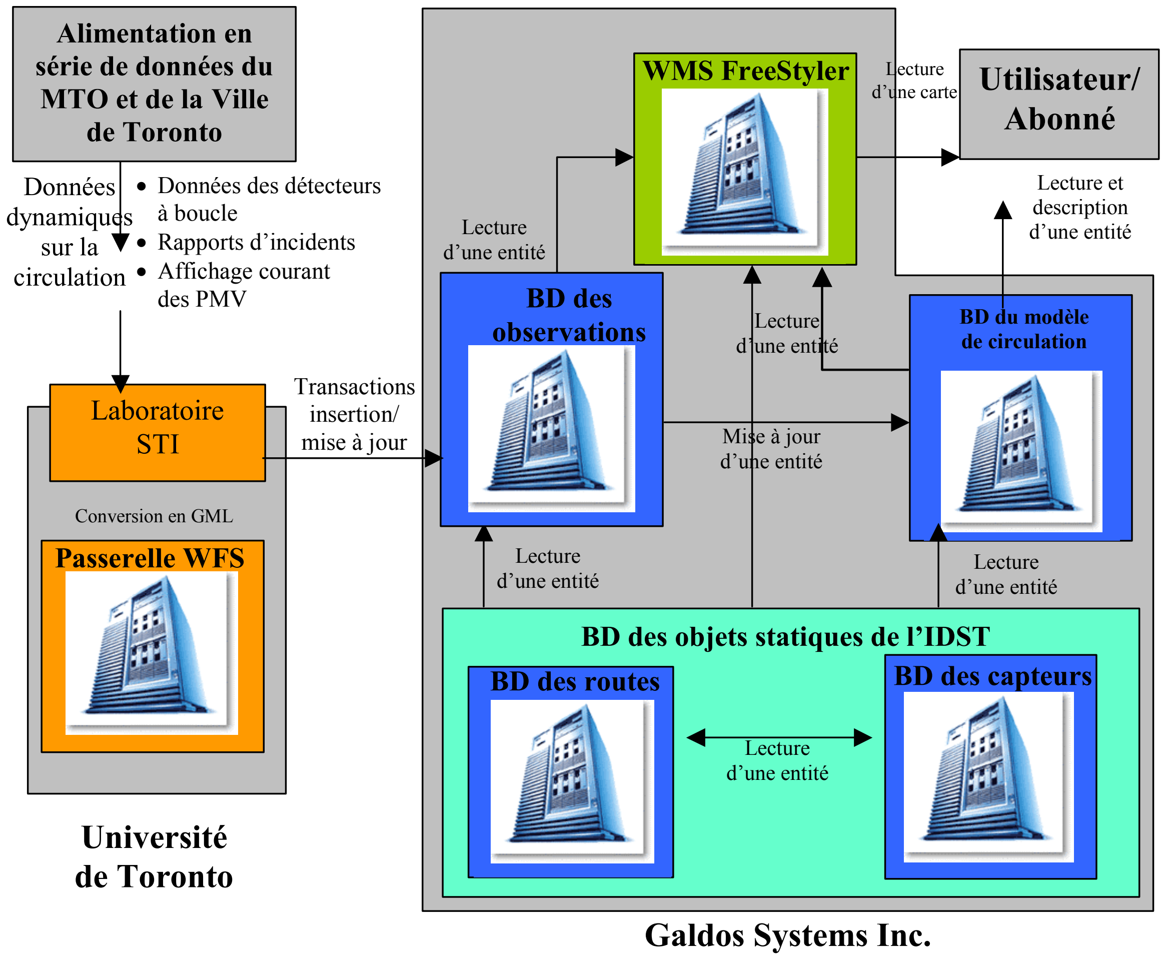


Figure 1 - Aperçu de l'architecture du prototype de l'IDST

La passerelle du service d'entités Web est située à l'Université de Toronto et est composée de scripts PHP développés par le laboratoire STI de l'Université de Toronto pour convertir en observations GML les données en temps réel du MTO et de la Ville de Toronto. Des demandes de transactions d'insertion sont alors créées (la charge étant le fichier XML nouvellement créé) et encadrées des balises `<wfs:transaction>` appropriées. Des catégories Java développées par Galdos Systems Inc. soumettent automatiquement la transaction WFS Insertion sur Internet au moyen du protocole http post.

Les données sur la circulation sont acheminées au laboratoire STI de l'Université de Toronto par deux ports série débitant à 19,2 kilobauds, un pour le MTO et l'autre pour la Ville de Toronto. L'interface physique est sérielle (RS-232); une connexion série «virtuelle» de plusieurs kilomètres est assurée par le système à fibre optique OTN de Siemens. Les données entrantes sont en format binaire, définies par des C structs, puis sérialisées. Une application lit ensuite les données arrivant au laboratoire STI et les entrepose dans une base de données MySQL locale.

Les lectures des détecteurs à boucle sont la source principale de données dynamiques de circulation reçues du MTO et de la Ville de Toronto par le prototype d'IDST. Les détecteurs à boucle sont situés sur chaque voie des autoroutes de la région de Toronto, à environ tous les demi-kilomètres. Les lectures sont faites aux vingt secondes; les paramètres mesurés sont le débit de véhicules, l'intensité de détection et, dans la plupart des cas, la vitesse moyenne.

Les lectures des détecteurs à boucle sont converties en GML et sont codées sous un type spécial d'entité appelée *LoopDetectorObservation* (Observation de détecteur à boucle*)*. Les valeurs d'attribut de cette entité comprennent la vitesse, le débit de véhicules et l'intensité de détection. Toutes les autres entités «transports» qui dépendent de ces lectures (p. ex., l'entité *Link* – ou Circulation) sont créées ou mises à jour automatiquement au fur et à mesure qu'une *LoopDetectorObservation* est reçue par le WFS Observations. L'entité *Link* représente un tronçon de l'autoroute associé à un ensemble de capteurs (détecteurs à boucle) qui couvrent toutes les voies dans une direction. L'entité contient la longueur du tronçon ainsi que la vitesse moyenne et le temps qu'il faut pour parcourir le tronçon en question; la vitesse moyenne et le temps de parcours sont établis à partir des observations des détecteurs à boucle. Une composante «Auditeur» du WFS Observations suit le registre des transactions et exécute une transaction «Insertion d'une entité *Link*» dans le WFS *Modèle de circulation* lorsqu'une *LoopDetectorObservation* est insérée dans le WFS Observations.

Les rapports d'incidents sont enregistrés par des opérateurs humains et respectent un format spécifique de fichier. La Ville de Toronto et le MTO ont chacun leur propre format de rapports d'incidents et les deux formats sont acceptés par le schéma d'application GML pour les STI.

Le MTO et la Ville de Toronto fournissent l'affichage courant de 40 panneaux à message variable (PMV) disposés le long des autoroutes de la région de Toronto. Lorsque le laboratoire STI reçoit l'affichage d'un PMV, la passerelle de WFS le convertit en une entité *Affichage PMV* codée en GML et soumet une demande de transaction WFS «Insertion/Mise à jour» à la base de données d'observations de la circulation (WFS) chez Galdos Systems Inc. Les utilisateurs de l'IDST desservis par un WFS peuvent recevoir l'*Affichage PMV* courant en l'extrayant de la base de données Observations (WFS) ou encore en le reproduisant automatiquement, comme dans le cas d'un rapport d'incident.

Le schéma d'application de l'IDST est composé de quatre fichiers XSD, organisés de la façon indiquée à la figure 2. Les flèches continues indiquent l'inclusion du schéma cible dans la source et les flèches en pointillé représentent l'importation du schéma cible par la source.

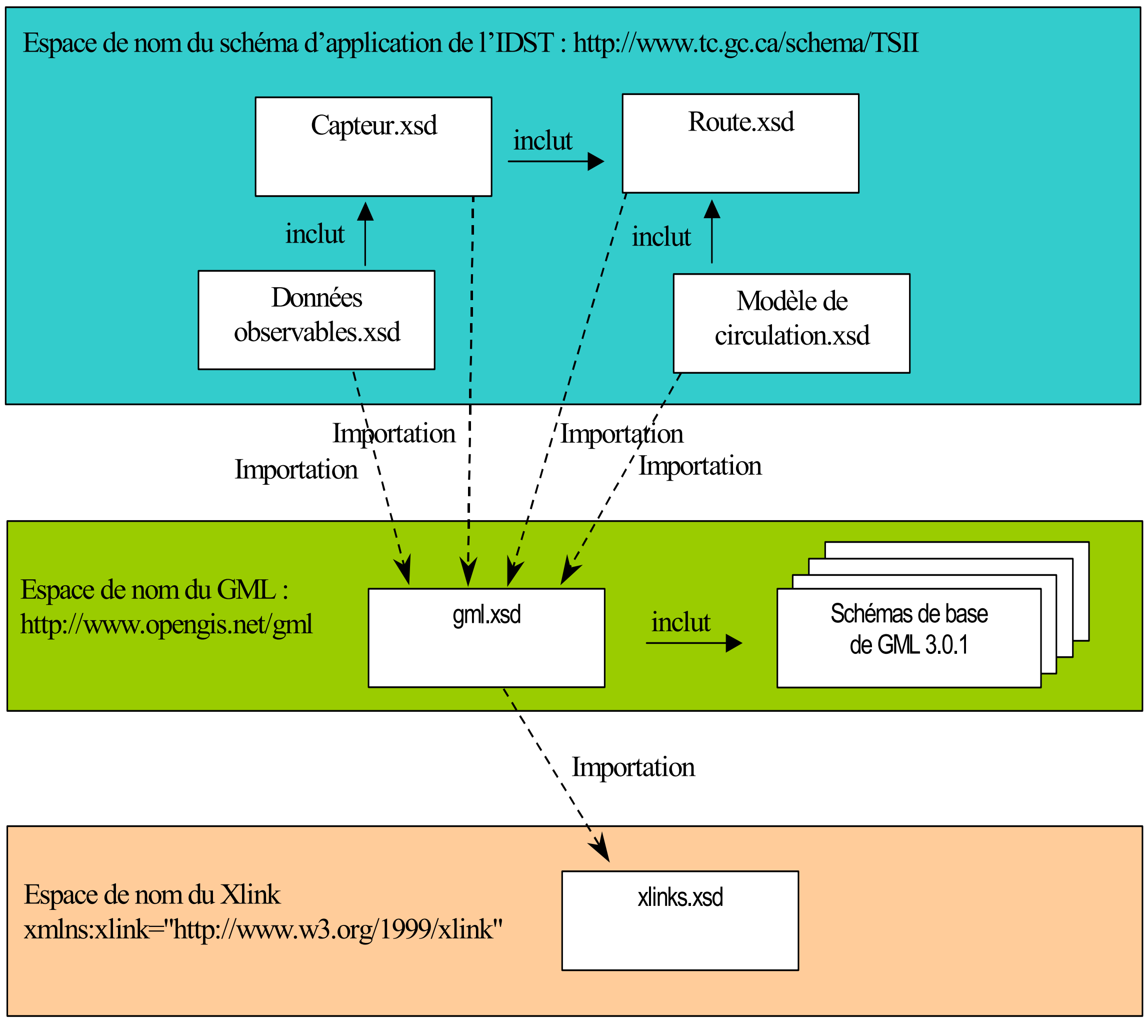

Espace de nom du schéma d'application de l'IDST : http://www.tc.gc.ca/schema/TSII
Capteur.xsd
inclut
Route.xsd
inclut
inclut
Données observables.xsd
Modèle de circulation.xsd
Importation
Importation
Importation
Importation
Espace de nom du GML : http://www.opengis.net/gml
gml.xsd
inclut
Schémas de base de GML 3.0.1
Importation
Espace de nom du Xlink xmlns:xlink="http://www.w3.org/1999/xlink"
xlinks.xsd

Figure 2 - Organisation du schéma de l'IDST

Chacun des quatre schémas est entreposé dans le WRS et dans chacun des serveurs d'entités Web correspondants : le WFS Observations de la circulation, le WFS Modèle de circulation, le WFS Routes et le WFS Capteurs.

Résultats

Le projet a consisté essentiellement à combiner des données statiques sur les routes et des lectures en temps réel de détecteurs à boucle en un scénario de circulation simple, et à mettre en oeuvre plusieurs des composantes de base de l'IDST en utilisant les quatre WFS, deux WMS et la passerelle WFS. Des extensions de logiciels de WFS ont été nécessaires pour permettre la mise à jour automatique des entités dynamiques. En particulier, la composante Auditeur/Gestionnaire d'actions de Cartalinea (l'application WFS de Galdos) a été développée dans le cadre de ce projet. La passerelle de WFS produit des *LoopDetectorObservation* qui sont codées en GML à partir des lectures en temps réel des capteurs du MTO et de la Ville de Toronto. Elle accepte également les *IncidentReports* (Rapports d'incidents) et les *MessageSignDisplays* (Affichages PMV) courants. Le client du WMS Galdos affiche une carte routière demandée à un autre WMS d'Intergraph, conforme aux normes de l'OGC, et visualise simultanément les entités de l'IDST, y compris les congestions routières, illustrées par des codes couleur sur la carte affichée par un fureteur Internet standard.

L'écran du WMS montré à la figure 3 présente les entités de l'IDST en tant que SVG superposées à une carte routière provenant d'un WMS d'Intergraph. Il en résulte un diagramme simple du mouvement de la circulation. Lorsqu'un événement onMouseOver est détecté sur une entité de l'IDST, une boîte de texte HTML Tooltips apparaît et indique certains attributs GML de l'entité en JavaScript. Lorsque la souris chevauche une entité *Route*, seul le nom de la route (valeur d'attribut *gml:name*) s'affiche, comme le montre la figure 3.

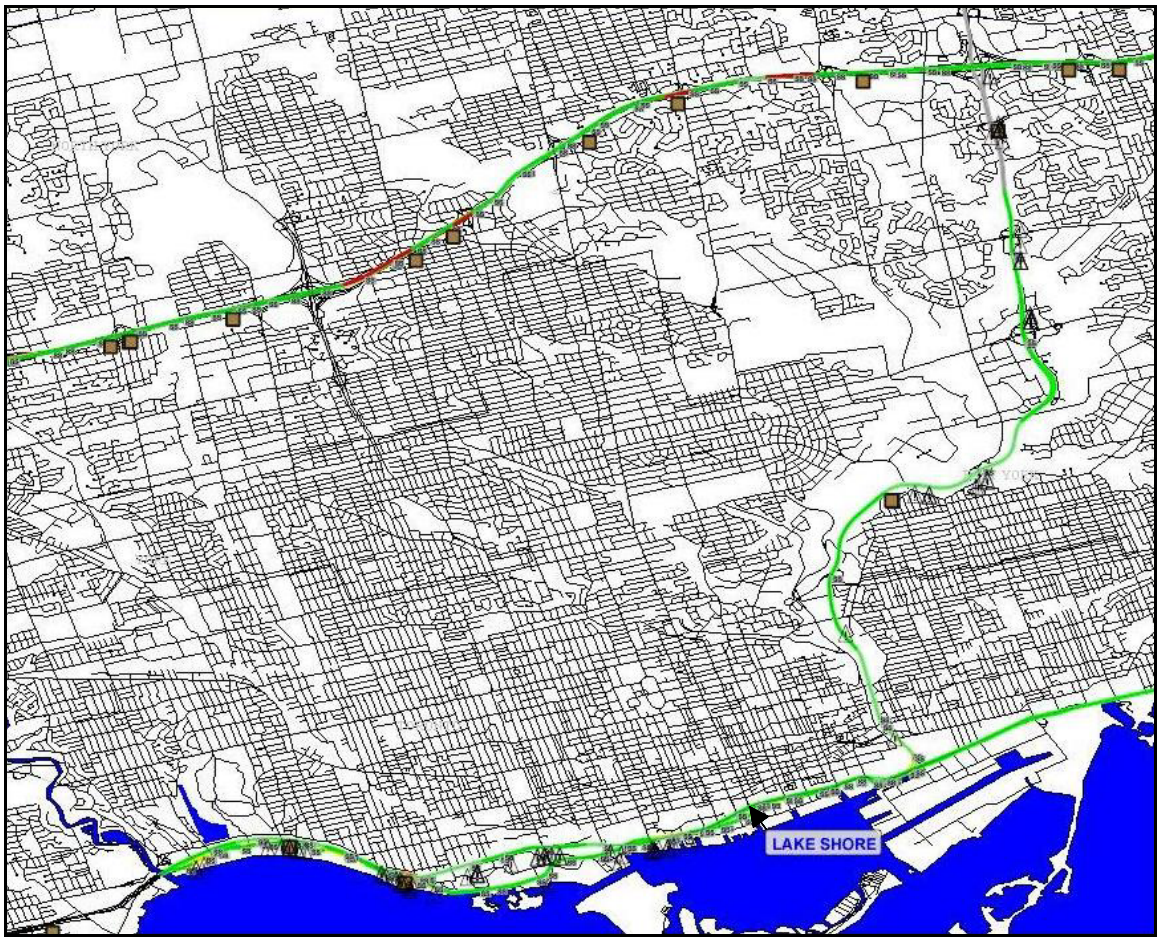


Figure 3 - Image d'une entité IDST avec le WMS FreeStyler

Les entités *Link* (Circulation) sont des tronçons de l'autoroute qui sont codés par couleur, du vert au rouge, selon le tableau 3. Le schéma des codes couleur s'inspire de celui adopté par le système de gestion de la circulation COMPASS du MTO, selon lequel l'une ou l'autre colonne est affichée selon un ordre de priorité. Ainsi, lorsque la vitesse moyenne est supérieure à zéro, la couleur de la cellule correspondant à la vitesse sert à établir le style du symbole *Link* (état de la circulation). Mais si la vitesse moyenne est nulle, c'est la colonne débit de véhicules qui sert à déterminer la couleur. Enfin, si le débit de véhicules est nul, c'est la colonne intensité de détection (pourcentage des intervalles de 20 secondes pendant lequel le détecteur à boucle est déclenché) qui sert à déterminer la couleur. Dans le cas où aucune lecture n'est transmise par le capteur au cours de l'intervalle de 20 secondes, la valeur par défaut est «-1» pour la vitesse moyenne, le débit de véhicules et l'intensité de détection, et le code couleur correspondant est le gris.

Tableau 3 - Schéma de codes couleur pour les entités *Link*

Vitesse moyenne (km/h)	Débit de véhicules (compte de véhicules/20 s/voie)	Intensité de détection (%)
> 100	> 0 et < 4	0 - 6
75 - 100	4 - 8	6 - 14
50 - 75	8 - 12	14 - 22
25 - 50	12 - 16	22 - 32
> 0 et < 25	> 16	> 32
-1	-1	-1

Au passage de la souris, la boîte de texte Tooltips énumère les valeurs des attributs suivants de l'entité *Link* :

- gml:id (identificateur unique)
- référence temporelle (valeur date/heure du schéma XML)
- vitesse moyenne (en km/h)
- intensité de détection (%)
- débit de véhicules (nombre de véhicules/20 s/voie)
- temps de parcours moyen (temps moyen en secondes prévu pour parcourir le tronçon représenté par l'entité *Link*)

Analyse

Le rendement global du prototype de l'IDST peut être mesuré par le délai entre le moment où un capteur prend une lecture et l'instant où les données sont affichées sur un écran. Il est à noter que chaque serveur utilisé dans le prototype d'IDST est un serveur Windows 2000 utilisant deux microprocesseurs Xeon à 2,8 GHz, à SDRAM DDR de 2GB. L'ensemble du cheminement des données est montré à la figure 4 et il est décomposé en une séquence de sept événements indiqués sur le diagramme. Le tableau 4 décrit brièvement chaque événement et en indique la durée moyenne.

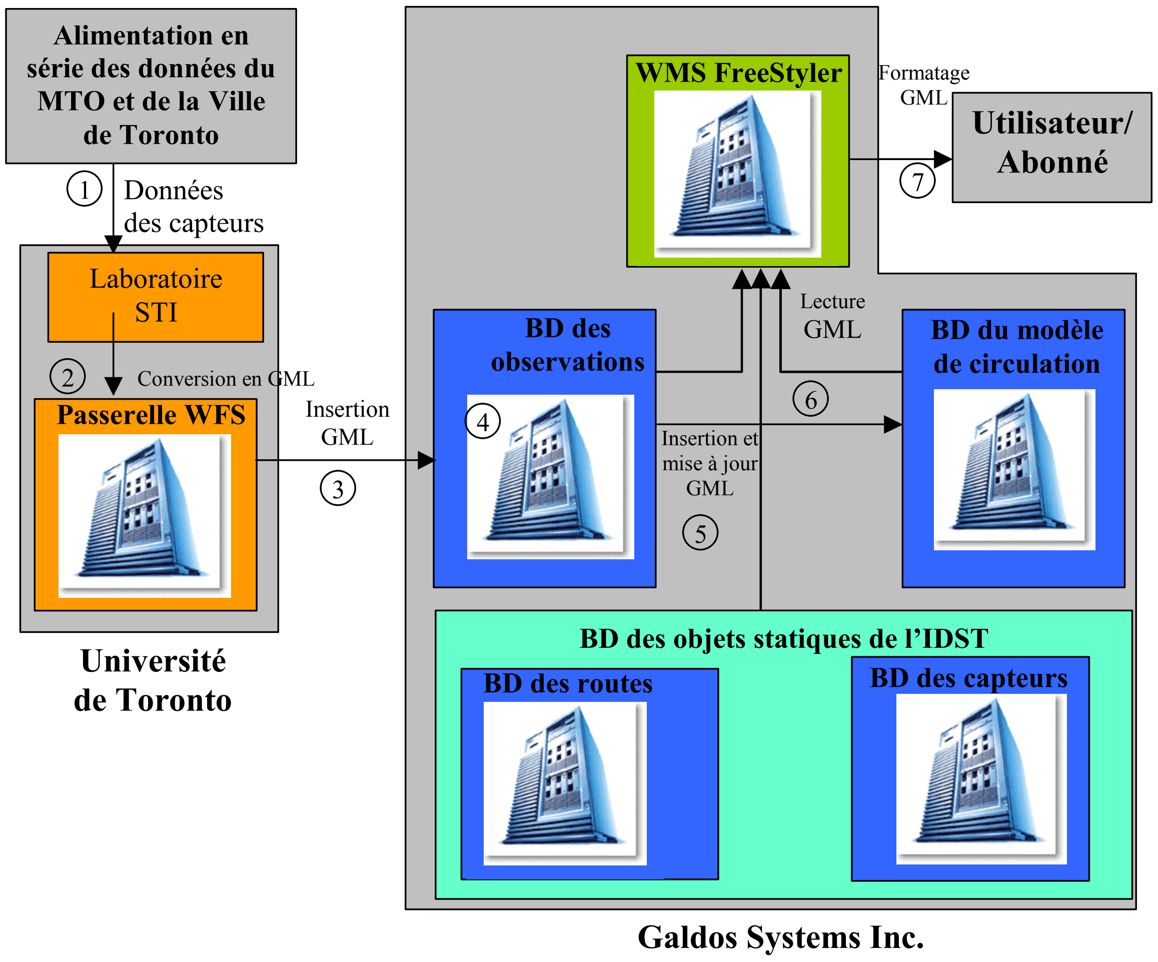


Figure 4 - Décomposition du cheminement des données

Le premier événement mesuré consiste en la préparation et la livraison des données des capteurs au laboratoire STI. Dans le cas des détecteurs à boucle, cet événement se décompose comme suit :

- Un véhicule passe sur un détecteur à boucle noyé dans la chaussée, créant une perturbation du courant électrique qui est détectée par le dispositif avancé de contrôle de la circulation (ATC, pour *Advanced Traffic Controller*) placé en bordure de la route.
- Chaque ATC recueille les données (débit, intensité de détection et vitesse) transmises pendant un intervalle de 20 secondes par plusieurs groupes de détecteurs à boucle appelés postes de détection de véhicules (VDS, pour *Vehicle Detector Stations*).
- Un serveur VAX au MTO reçoit et entrepose les données de tous les ATC qui bordent l'autoroute.
- Un processus intégré au serveur VAX prend une lecture de la base de données toutes les secondes et surveille les changements. Lorsqu'une mise à jour est détectée, les données courantes sont lues et comprimées (temps requis : 2,5 secondes en moyenne).

- Les données comprimées (en moyenne 10 kilo-octets) sont transmises par un lien en série à un débit de 19,2 kilo-octets/s au laboratoire STI de l'Université de Toronto (temps de transmission : en moyenne 11 secondes).
- Les données reçues par le laboratoire STI sont décompressées puis moyennées pour toutes les voies et chaque VDS, avant d'être insérées dans une base de données MySQL (temps requis : en moyenne 0,9 seconde).
- Un processus distinct prend une lecture de la base de données du MTO toutes les cinq secondes, vérifiant s'il y a eu des mises à jour de l'information sur les PMV et les incidents. Lorsqu'une mise à jour est détectée, les données sont immédiatement envoyées au laboratoire STI.

La durée totale de l'événement «préparation et transmission des données des capteurs» est résumée à la première ligne du tableau 4. Les durées respectives de chacun des six autres événements du cheminement des données sont aussi indiquées au tableau 4. Le cas échéant, la technologie de traitement ou le protocole principal utilisé pour chaque événement est indiqué entre parenthèses, (p. ex., HTTP POST).

Tableau 4 - Fonctionnement du prototype de l'IDST

ÉVÉNEMENT	DESCRIPTION	DURÉE MOYENNE (secondes) Données du MTO	DURÉE MOYENNE (secondes) Données de la Ville de Toronto
1	Préparation et transmission à la base de données du laboratoire STI des données provenant des capteurs	20,0	18,00
2	Conversion en GML (script PHP)	0,1	0,09
3	Envoi d'une requête via la passerelle WFS (HTTP POST) y compris les délais du réseau	1,6	1,10
4	Engagement de la base de données Observations (base de données XHIVE)	171,0	110,50
5	Mise à jour des données sur la circulation (auditeur/gestionnaire d'actions sur Java). Cet événement est concurrent de l'événement 4.	332,5	209,00
6	Requête Lecture d'une entité (HTTP LIRE)	5,5	5,00
7	Conversion de GML à SVG (XSLT)	2,1	2,00
	Temps total	**361,8**	**235,19**

Conclusion

Le développement du prototype de l'IDST, fondé sur le GML et sur les services Web de l'OGC démontre qu'une solution à norme ouverte peut accepter des données dynamiques provenant de capteurs de circulation. Une telle solution offre des avantages évidents, notamment l'interopérabilité, mais le rendement, bien qu'acceptable, pourrait être amélioré. Les résultats touchant le rendement indiquent un délai d'environ six minutes entre le moment où les capteurs prennent la lecture et l'affichage, pour ce qui est des données du MTO, et de quatre minutes, pour ce qui est des données de la Ville de Toronto. Le temps a manqué pour examiner plus en profondeur des moyens de réduire les délais, par exemple en optimisant la configuration des logiciels, ce qui aurait pu améliorer le rendement.

Le client WMS utilisé aux fins de l'étude récupérait et affichait les cartes SVG de deux serveurs de cartes Web distincts, dont les données d'entités sous-jacentes provenaient de bases de données différentes et étaient stockées sous des formats différents. Le WMS hébergé chez Galdos Systems faisait des requêtes d'entités en GML à quatre WFS à bases de données sous-jacentes XHIVE (XML natif), tandis que le WMS hébergé chez Intergraph récupérait les données d'une base de données ArcSDE/Oracle. La mise en œuvre de l'IDST a donc révélé une intégration, une distribution et un affichage sans coupure de données de transport provenant de plusieurs sources et stockées sous des formats différents, ce qui rend possible une infrastructure «prête à utiliser» et à couverture étendue de données sur les transports. D'autres sources de données statiques et dynamiques (capteurs de conditions météorologiques, appareils GPS, etc.), réunies dans n'importe quelle base de données, peuvent être reliées à l'IDST via un WFS. L'ensemble élargi de données sur les transports ainsi constitué peut alors être affiché sur la même carte ou récupéré en un même ensemble de résultats pour traitement. L'IDST fondé sur une solution à norme ouverte de l'OGC ouvre donc la voie à un système prêt à utiliser et à couverture étendue pour les données en transports.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ATC	Advanced Traffic Controller
CMS	Changeable Message Sign
CRS	Coordinate Reference System
GIS	Geographic Information System
GML	Geography Markup Language (OGC specification)
HTTP	Hyper Text Transfer Protocol
ISO	International Organization for Standardization
ITS	Intelligent Transportation System
MTO	Ministry of Transportation of Ontario
OGC	Open Geospatial Consortium
PHP	Hypertext Preprocessor (an open source scripting language)
SLD	Styled Layer Descriptor
SVG	Scalable Vector Graphics
TMD	Traffic Model Database
TOD	Traffic Observation Database
TSII	Transportation Spatial Information Infrastructure
VDS	Vehicle Detector Station
WFS	Web Feature Service (OGC implementation specification)
WMS	Web Map Service (OGC implementation specification)
WPS	Web Processing Service (OGC draft implementation specification)
WRS	Web Registry Service (An ebRIM profile of the OGC Catalogue Service)
XML	eXtensible Markup Language
XSD	XML Schema Definition
XSLT	eXtensible Stylesheet Language Transformation

1 INTRODUCTION

The focus of this R&D project was to develop an ITS spatial infrastructure, based on emerging spatial standards from the Open Geospatial Consortium (OGC) and the World Wide Web Consortium, and in a manner consistent with the Canadian ITS Architecture and Data Element Registry. The OGC is an international standards organization that is leading the development of standards for geospatial and location-based services. Through member-driven consensus programs, the OGC works with government, private industry, and academia to create open and extensible Web services and software application programming interfaces for geographic information systems (GIS). The OGC Web services used in this R&D project use a standard data transfer format, Geography Markup Language (GML), soon to be an ISO standard (ISO/TC 211 19136) for the encoding of geographic information.

GML is an eXtensible Markup Language (XML) application that is fast becoming the world standard for geographic information delivery over the Internet. Using GML as a data transfer format, geographic information can be integrated from multiple application domains, regardless of what the underlying data storage formats are. This integrated data can then be delivered via web service transactions as distinct features and displayed in a Web browser using any selected map style.

The main goal of this project was to develop a prototype Transportation Spatial Information Infrastructure (TSII) that employs two OGC Web Map Services (WMS) to deliver real-time traffic data as a Scalable Vector Graphics (SVG) map to the end user. By refreshing the SVG map, updated readings from traffic sensors are pulled from a Web Feature Service (WFS) and redisplayed on a colour-coded traffic congestion map. A Web Registry Service (WRS) is an online web service used to discover the component WFSs, WMSs and GML data sets of the TSII. More specifically, the TSII comprises the following:

- A GML Application Schema for ITS based on the Canada ITS Data Element Registry, representing a simplified traffic model chosen to illustrate the handling and integration of dynamic traffic information;
- OGC-compliant implementations of a WFS to integrate and distribute geographic information encoded as GML;
- OGC-compliant implementations of a WMS used to style and display GML data received from a WFS;
- An OGC-compliant implementation of the ebRIM profile of the Catalogue for the Web, otherwise known as a WRS used to discover Web services and GML datasets related to the TSII;
- A WFS Gateway, which converts real-time Ontario Ministry of Transportation (MTO) and City of Toronto sensor data to GML and transfers this to a WFS.

Galdos Systems provided its own implementations of these web services for use in this project, namely Cartilinea (WFS), FreeStyler (WMS), and Indicio (WRS/Catalogue).

The aim was to develop a prototype TSII complex enough to capture the information necessary to display a traffic flow map similar to the MTO COMPASS traffic flow map in Figure 1-1.

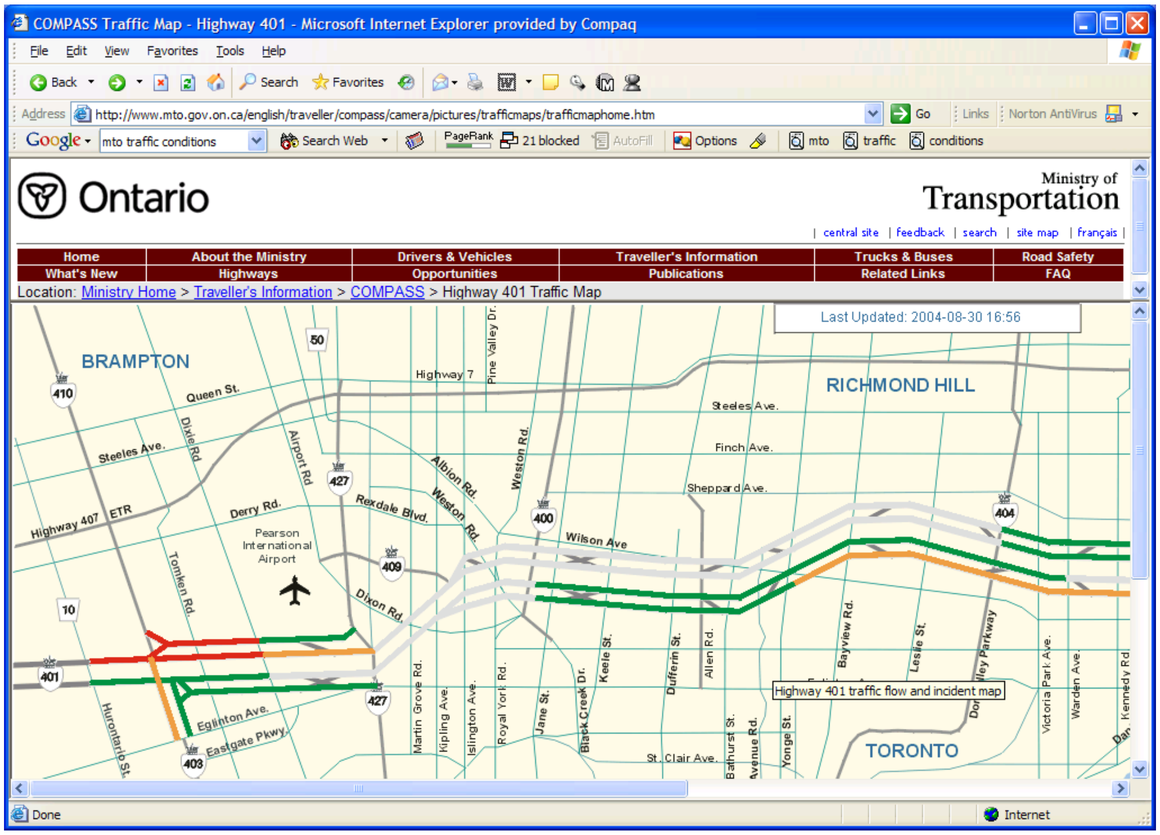

COMPASS Traffic Map - Highway 401 - Microsoft Internet Explorer provided by Compaq
http://www.mto.gov.on.ca/english/traveller/compass/camera/pictures/trafficmaps/trafficmaphome.htm
Ontario
Ministry of Transportation
Home
About the Ministry
Drivers & Vehicles
Traveller's Information
Trucks & Buses
Road Safety
What's New
Highways
Opportunities
Publications
Related Links
FAQ
Location: Ministry Home > Traveller's Information > COMPASS > Highway 401 Traffic Map
Last Updated: 2004-08-30 16:56
BRAMPTON
RICHMOND HILL
TORONTO
Pearson International Airport
Highway 401 traffic flow and incident map
Done
Internet

Figure 1-1 MTO COMPASS Traffic Flow Map

One of the main advantages of adopting an open standard solution over a proprietary one is interoperability, which leads to lower overall costs of maintaining data. Interoperability is demonstrated in this project by integrating static and dynamic transportation data from multiple sources. The sources of data in this project are as follows:

- Road and highway data, including geometry and nonspatial metadata obtained from NAVTEQ, a well-known data supplier worldwide;
- Road and waterbody map data from an ArcSDE/Oracle database delivered via an OGC-compliant WMS hosted by the GIS vendor, Intergraph;
- Loop detector data from MTO, including identification, functionality, real-time sensor readings, and location along Highway 401 in the metro Toronto area;
- Changeable Message sign information from MTO, including real-time message display, identification, and location along Highway 401 in the metro Toronto area;
- Loop detector data from the City of Toronto, including identification, functionality, real-time sensor readings, and location along Don Valley Parkway, Gardener Expressway, and Lake Shore Drive;

- Changeable Message sign information from the City of Toronto, including real-time message display, identification, and location along Don Valley Parkway, Gardener Expressway, and Lake Shore Drive;
- Traffic incident reports from MTO;
- Traffic incident reports from the City of Toronto.

Note that the TSII enables other sources of static and dynamic data (weather sensors, GPS devices, etc.) to be integrated with the existing traffic data, which could then be displayed with the traffic information on the same map.

2 APPROACH

In the first phase of this R&D project, an extensive literature review was conducted as summarized in Section 2.1.

2.1 Literature Review

The articles, books, XML vocabularies, and other documentation that were studied in the literature review phase of this project are as follows:

- TMDD Center-to-Center Concept of Operations and Requirements Standard [1],
- Geographic Data Files (GDF) [2],
- ITS Architecture for Canada [3],
- MTO ITS Internet Applications: Data Exchange Standards Report [4],
- MTO ITS Internet Applications: Web Site Review Report [5],
- TransXML [6],
- TranXML [7],
- TourML [8],
- Open Location Services (OLS) [9],
- XML in Transportation [10],
- Architecting with RM-ODP [11],
- Modelling the Impact of ITS on Surface Goods Movement in Canada [12],
- Study on ITS Applications within the Canadian Trucking Industry [13].

The literature review was followed by the requirements analysis phase of this project, in which the following tasks were performed:

- XML grammars related to transportation were considered for use in the TSII and a program of co-operation was intitiated with an XML-based transportation project in the U.S.A. called TransXML;
- GIS use cases from ITS practice were listed and a subset of these were targeted for implementation in this project;
- Candidate classes from the ITS Architecture for Canada were selected for implementation.

The following subsections provide the results and relevant details of the above tasks.

2.1.1 XML Grammars for Transportation

The XML grammars related to transportation that were reviewed and considered for use in this project were:

- TranXML
- TransXML
- TourML
- GDFXML

A brief summary of the findings for each XML grammar is given below under the corresponding heading.

TranXML

TranXML was originally developed in 2001 by Transentric (www.transentric.com) and has been maintained by the Open Applications Group (OAGI) as of November 2002. TranXML is an XML solution being proposed as an industry standard for e-Commerce related transactions between shippers and carriers. It was created specifically to serve needs surrounding the procurement and delivery of transportation and logistics services required for supply chain execution. TranXML is intended to supplement the extensive investment in EDI (Electronic Data Interchange—an ANSI standard for transportation) that has powered the transportation industry for many years. Most of the elements defined in the TranXML set of schemas are specific to commercial motor carrier transactions and are not entirely relevant to this TSII project. The schema elements defined by TranXML are summarized at www.transentric.com.

TransXML

The U.S. TransXML project that started in 2004 is led by Cambridge Systematics Inc. and funded by the U.S. National Cooperative Highway Research Program (NCHRP). The main objective of this project is to develop broadly accepted XML schemas for exchange of transportation data. Note that the TransXML project excludes ITS, and so has limited overlap with this TSII project. Galdos extended an offer to Cambridge Systematics to cooperate with the TransXML project team and this offer was accepted. The result of this cooperation was that GML was adopted as the standard baseline for the TransXML schemas.

TourML

Tourism markup language (TourML) developed in 2003 by the Open Tourism Consortium (OTC) is an XML encoding for the transport and storage of tourism information. TourML defines abstract objects, events of interest and (roughly 65) concrete tourist schema objects such as Hotels, Restaurants, MovieTheatres, Banks, etc. in the main schema file 1-0.xsd. TourML is extensible in the sense that schema elements have named complex types enabling the creation of application schemas that can define extensions or restrictions to the existing types. The elements and types in TourML are beyond the scope of this TSII project, but are recommended for consideration in future extensions to this project.

GDFXML

GDFXML is an XML application schema for Geographic Data Files (GDF). In GDFXML, generic feature classes are defined, such as PointFeature, LineFeature and AreaFeature, each having a geometry property (Point, Line, or Area) as indicated by the name. The attributes of each of these classes are uniform for all features of that geometry type. Feature and attribute codes are then checked against feature and attribute catalogues to determine the feature type and its properties, respectively. This approach to the definition of features and attributes in GDFXML goes against the spirit of the feature model of GML, where concrete features and their properties are meant to be defined using semantically meaningful element names in application schemas. Although the feature models of GDFXML and GML differ, the relevant features (roads, highways) in the transportation theme of the GDF feature catalogue are modelled in GML (see Appendix A) for use in this project.

2.1.2 GIS Use Cases in ITS Practice

This section presents a summary of the GIS use cases employed or needed in current ITS practice. A list of these GIS use cases is provided here:

a. Discover and retrieve transportation/service objects by spatial queries and/or attribute filter
b. Share and manage traffic information, including incidents/events, traffic/road conditions, vehicle operational status
c. Status and control of traffic devices
d. CRS conversion and management
e. Map production/maintenance
f. Commercial/transit vehicle management
g. Route guidance

Most of the use cases listed above can be handled directly by combining WFS, WMS, and WRS transactions. The exception is use case g (route guidance), which can be handled by a Web Processing Service, currently an OGC draft implementation specification. The targeted use cases for this prototype are the first five (a to e) above.

2.1.3 Physical Architecture of ITS Architecture for Canada

The Canadian ITS Physical Architecture is organized into four classes: Travellers, Centres, Vehicles, and Wayside. In this R&D project a representative subsystem from each of four classes will be implemented. Figure 2.1-1 illustrates the classes and subsystems of the Canadian ITS Physical Architecture.

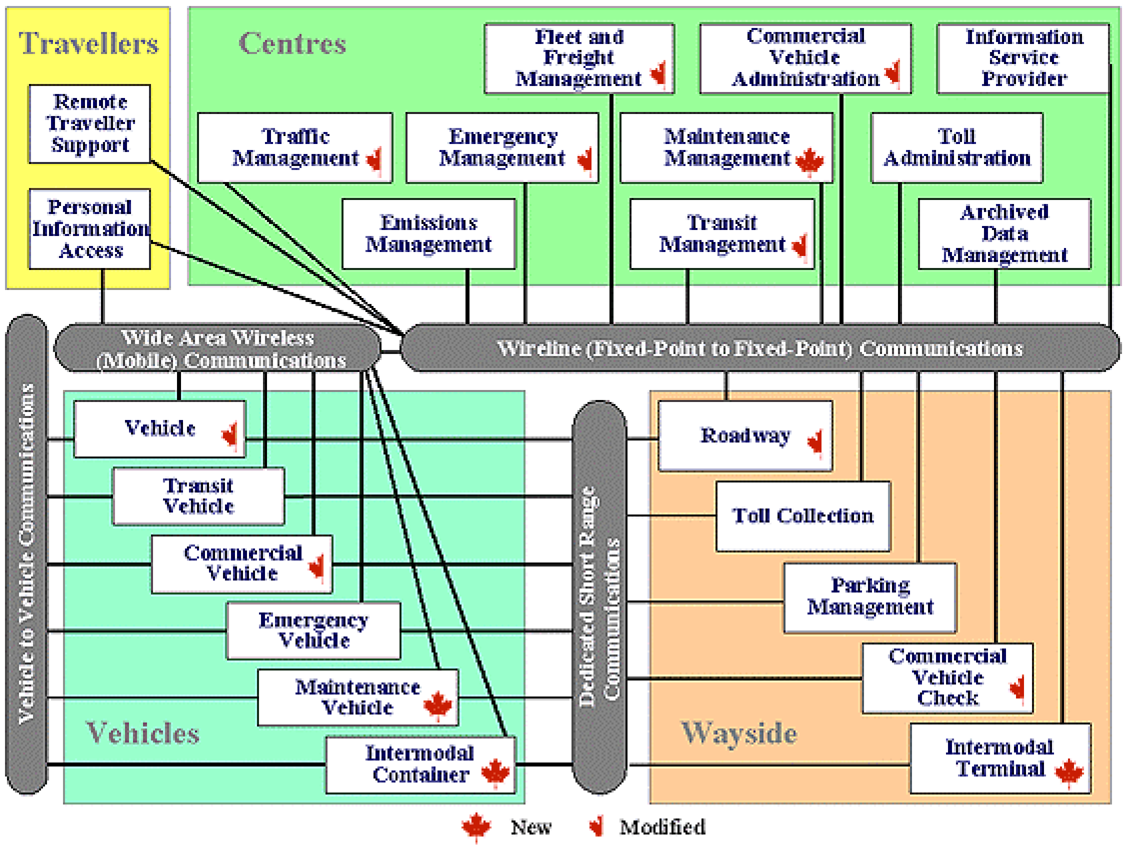


Figure 2.1-1 Overview of the Canadian ITS Physical Architecture V1.1

Targeted Subsystems

The subsystems targeted for implementation in this project are based on the summary of GIS use cases listed in Section 2.1.2 and the available traffic sensor data from Toronto area highways at the University of Toronto ITS Lab (see Section 3.3). Table 3.1-1 lists the subsystems and the associated Process Specifications (PSpecs) that are implemented in the TSII.

Table 2.1-1 Subsystems and Process Specifications Implemented in TSII

Class	Subsystem	Process Specifications
Centres	Traffic Management	1.1.4.1-Retrieve traffic data, 1.1.2.2-Process traffic data, 1.3.4.1-Retrieve incident data, 1.3.3-Respond to current incidents
Wayside	Roadway	1.1.1.1-Process traffic sensor data, 1.6.1.1-Detect roadway events, 1.1.1.4-Manage data collection and monitoring
Vehicles	Vehicle	6.2.2-Prepare and output in-vehicle displays
Travellers	Remote Traveller Support	6.3.2-Inform traveller

2.2 TSII USE CASES

The traffic condition and incident reporting service provides facilities for systems and users to report traffic conditions using the WFS gateway by generating and submitting WFS transactions to the Traffic Model Database and Traffic Observation Database. The traffic incident reporting and traffic condition reporting use cases are detailed in Subsections 2.2.2 and 2.2.7. A user (e.g. traveller or dispatcher) of the TSII can retrieve traffic conditions and incidents by accessing the traffic model database and Traffic Observable Database. The traffic condition and incident retrieving use cases are detailed in Subsections 2.2.3, 2.2.6, and 2.2.8. A typical scenario that fits the sequence of use cases described in the following subsections is represented by the following sequence of events:

- A traffic incident, such as a car accident, occurs.
- The details of the traffic accident, including the time, location and roads affected, are reported.
- Notification of the traffic incident is broadcasted to subscribers including emergency services, highway maintenance, transit dispatchers, vehicle travellers (via changeable message signs, onboard WFS, radio), etc.
- A user of the TSII, such as any one of the subscribers listed above, requests the details of the traffic incident.
- Road sensors measuring mean speed and traffic volume collect data on traffic conditions, which are reported to a dynamic traffic model.
- The user requests the traffic conditions along a travel route, which may be affected by the traffic incident.
- The traffic incident and traffic conditions along the selected route are displayed to the user as a colour-coded map overlaid on a road map.

The following subsections describe the role of the actors (users and web services), a text description, and a corresponding diagram for each use case. It is assumed in the following use cases that an access control service is already in place that restricts access to the traffic data, based on some security policy. For example, users will only be able to create an incident report if they are authorized to access the WFS Gateway.

2.2.1 Actors

Human actors

Actor	Description
User	A traveller, dispatcher, traffic incident reporter or traffic incident subscriber

System actors

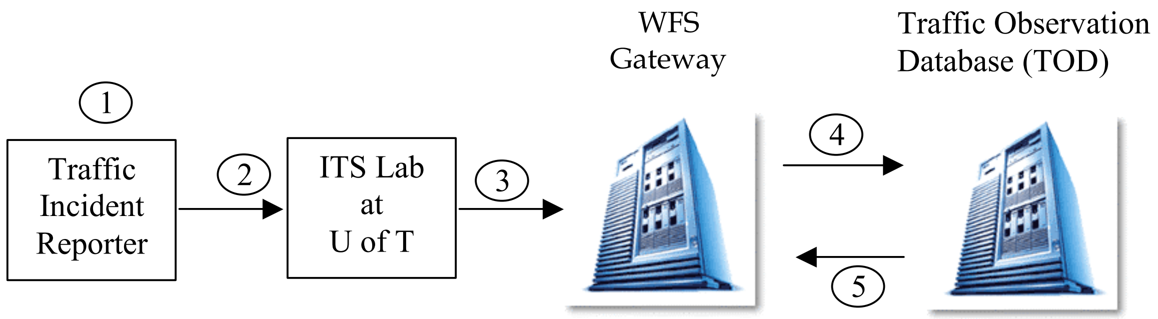
Actor	Description
WFS Gateway	Converts traffic sensor and incident data to WFS transactions (Insert, Update) to the dynamic traffic model and incident databases
Road Network Server	A WFS that stores and distributes static road network data representing the 400 series highways around Toronto
Traffic Model Database (TMD)	A WFS that stores and distributes traffic condition data
Traffic Observation Database (TOD)	A WFS that stores and distributes traffic observables, including loop detector, traffic incident report and CMS data
CRS Registry	A WRS that stores and allows for the discovery of CRS data (2D, and 1D Linear Reference Systems) and transformations
FreeStyler	A WMS that styles and displays GML data as a map
Listener	A component of a WFS that is configured to detect events and then trigger appropriate actions (e.g. replicate or create new features), which are specified by policies

2.2.2 Traffic Incident Reporting

Use case text description

Description	This use case describes how a traffic incident is reported to the *Traffic Observation Database* (*TOD*) WFS
Priority	High
Actors	1. *User* 2. *WFS Gateway* 3. *Traffic Observation Database (TOD) WFS*
Pre-conditions	1. The *User* must be authenticated and be authorized to complete a Traffic report.
Events	1. The *User* creates an incident report in a predefined format by filling in the available fields. 2. The report is submitted to the ITS Lab. 3. The report is converted to a GML `TrafficIncident` feature and submitted to the *WFS Gateway*. 4. The *WFS Gateway* creates and sends an Insert Feature transaction to *TOD*. 5. *TOD* responds with Transaction Results.
Alternate Paths	
Post-conditions	1. Upon successful insertion of `TrafficIncident`, the *TOD* reports (push replicates) the newly inserted traffic incident results to all subscribing services (WFSs).
Notes	Subscribers to traffic incident information may be any users of the transportation system with access to a WFS in the TSII, e.g. dispatchers, passengers, drivers, etc.

Use case diagram—Traffic Incident Reporting

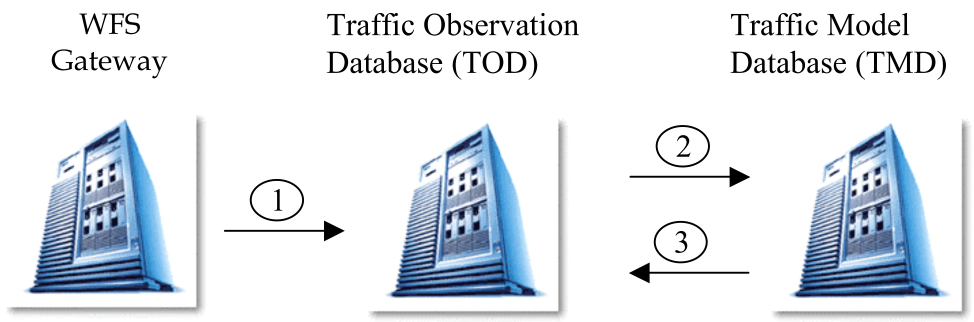


2.2.3 Traffic Incident Replication

Use case text description

Description	This use case describes how a traffic incident is replicated by the *Traffic Observation Database* (*TOD*) to the *Traffic Model Database (TMD)*
Priority	High
Release	1
Actors	1. *Traffic Observation Database (TOD)* 2. *Listener* 3. *Traffic Model Database (TMD)*
Pre-conditions	1. The *TOD* receives an Insert or Update transaction from the *WFS Gateway*
Events	1. The *Listener* detects an Insert *IncidentReport* in the transaction log of the *TOD*. 2. The *TOD* responds by performing the same Insert *IncidentReport* transaction to all subscribing WFSs (*Users*). 3. The subscribing WFS responds with Transaction Results.
Alternate Paths	
Post-conditions	1. Upon successful insertion of `TrafficIncident`, the *TOD* reports (push replicates) the newly inserted traffic incident results to all subscribing services (WFSs).
Notes	Subscribers to traffic incident information may be any users of the transportation system with access to a WFS in the TSII, e.g. dispatchers, passengers, drivers, etc.

Use case diagram—Traffic Incident Replication

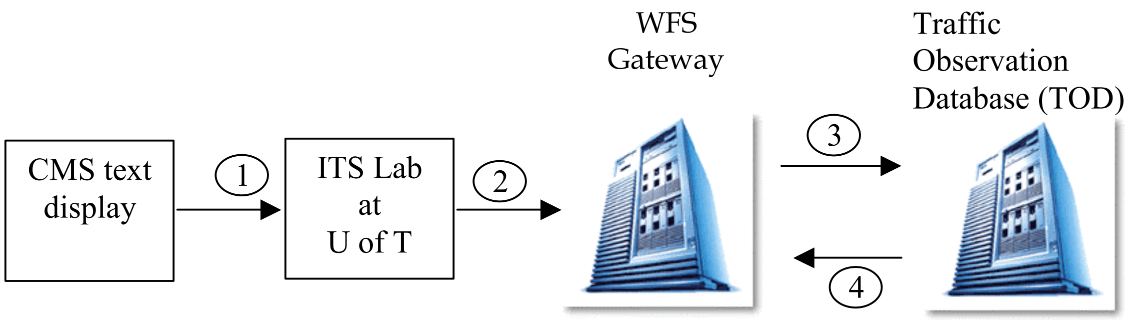

WFS Gateway
Traffic Observation Database (TOD)
Traffic Model Database (TMD)
1
2
3

2.2.4 Changeable Message Sign Reporting

Use case text description

Description	This use case describes how a CMS is reported to the *Traffic Obseration Database* (*TOD*)
Priority	High
Release	2
Actors	1. *WFS Gateway* 2. *Traffic Observation Database (TOD) WFS*
Pre-conditions	
Events	1. The current CMS text display is received by the ITS Lab 2. The CMS text display is converted to a GML `MessageSignDisplay` feature and submitted to the *WFS Gateway*. 3 The *WFS Gateway* creates and sends an Insert Feature transaction to *TOD*. 4. *TOD* responds with Feature Insert Results.
Alternate Paths	
Post-conditions	1. Upon successful insertion of `MessageSignDisplay`, the *TOD* reports (push replicates) the newly inserted `MessageSignDisplay` results to all subscribing services (WFSs).
Notes	Subscribers to traffic information may be any users of the transportation system with access to a WFS in the TSII, e.g. dispatchers, passengers, drivers, etc.

Use case diagram— Changeable Message Sign Reporting

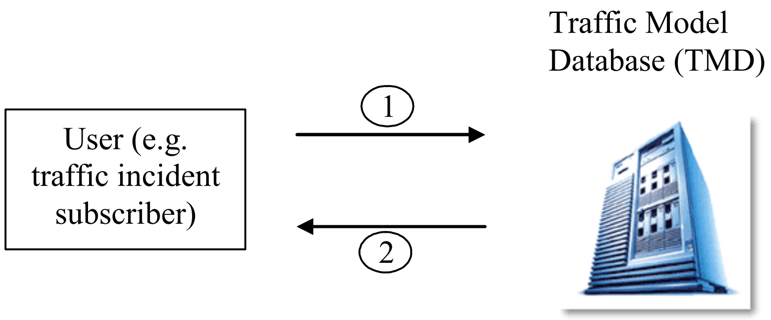


2.2.5 User Requests Traffic Incident Details

Use case text description

Description	This use case describes how a user (such as a traffic incident subscriber) with access to GDS requests the `TrafficIncident` Feature.
Priority	High
Release	2
Actors	1. *User* 2. *Traffic Model Database (TMD) WFS*
Pre-conditions	1. The *User* must be authenticated and authorized to access the *TOD*. 2. The *User* is either a traffic incident subscriber and receives the traffic incident notification or requests the traffic incident information (pull replication).
Events	1. The *User* submits a Get Feature request for `TrafficIncident` with the given ID to the *TMD*. 2. *TMD* returns the `IncidentReport` feature.
Alternate Paths	
Post-conditions	
Notes	

Use case diagram— User Requests Traffic Incident Details

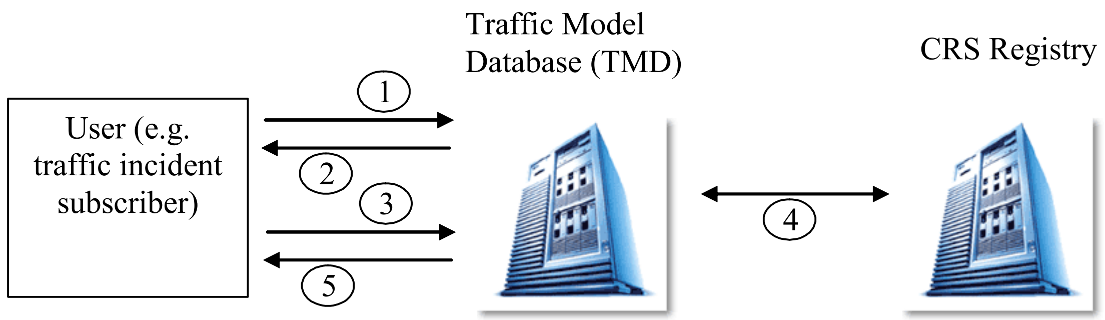


2.2.6 User Requests Traffic Incident Details in a Different CRS

Use case text description

Description	This use case describes how a user (e.g. traffic incident subscriber) with an onboard WFS requests the `TrafficIncident` with coordinate values in a specific CRS.
Priority	Medium
Release	2
Actors	1. *User* 2. *Traffic Model Database (TMD)* 3. *Coordinate Reference System (CRS) registry WRS*
Pre-conditions	1. The *User* must be authenticated and be authorized to access the *TMD*. 2. The *User* receives the traffic incident notification including the feature type (`IncidentReport`) and ID.
Events	1. The *User* submits a Get Feature request for `IncidentReport` with the given ID to the *TMD*. 2. *TMD* returns the `IncidentReport` feature with all coordinate values in the default CRS (e.g. linear reference system). 3. The *User* submits a more refined Get Feature request for `IncidentReport` with the given ID and CRS specified (e.g. 2D UTM projection). 4. *TMD* requests from *CRS registry* the coordinate conversion of all coordinate values from source to target CRS. 5. *TMD* returns `IncidentReport` feature with coordinate values in specified CRS.
Alternate Paths	
Post-conditions	
Notes	

Use case diagram— User Requests Traffic Incident Details in a Different CRS

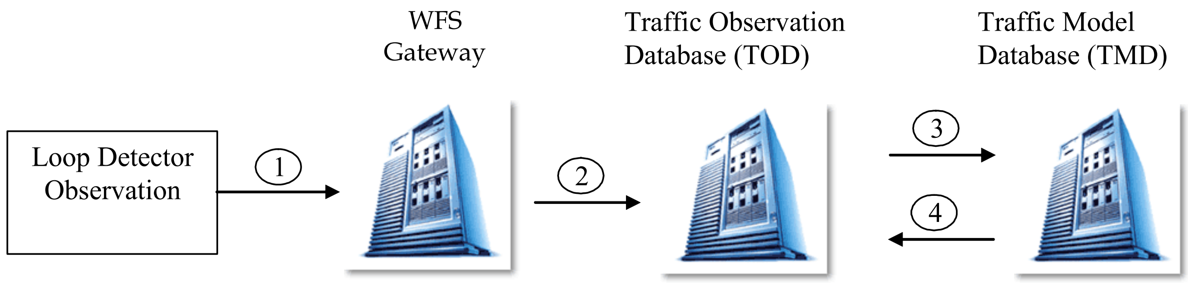

Traffic Model Database (TMD)
CRS Registry
User (e.g. traffic incident subscriber)
1
2
3
4
5

2.2.7 Traffic Condition Reporting

Use case text description

Description	This use case describes how traffic conditions as measured by road highway sensors are updated to the *Traffic Model Database* (*TMD*).
Priority	High
Release	1
Actors	1. *WFS Gateway* 2. *Traffic Model Database (TMD) WFS*
Pre-conditions	
Events	1. The road sensor data (mean speed, traffic volume, vehicle occupancy) is converted to a *LoopDetectorObservation* and submitted to *WFS Gateway*. 2. WFS Gateway Inserts *LoopDetectorObservation* to *TOD* 3. The *Listener* detects the inserted observation, creates the dynamic `Link` feature and insert or updates it to the *TMD*. 4. *TMD* responds with Insert/Update Feature Results.
Alternate Paths	
Post-conditions	1. Under specific conditions, the *TMD* may report (push replicate) the `Link` feature to all subscribing services (WFSs).
Notes	Subscribers to traffic model information may be any users of the transportation system with access to a WFS in the TSII, e.g. dispatchers, passengers, drivers, etc.

Use case diagram— Traffic Condition Reporting

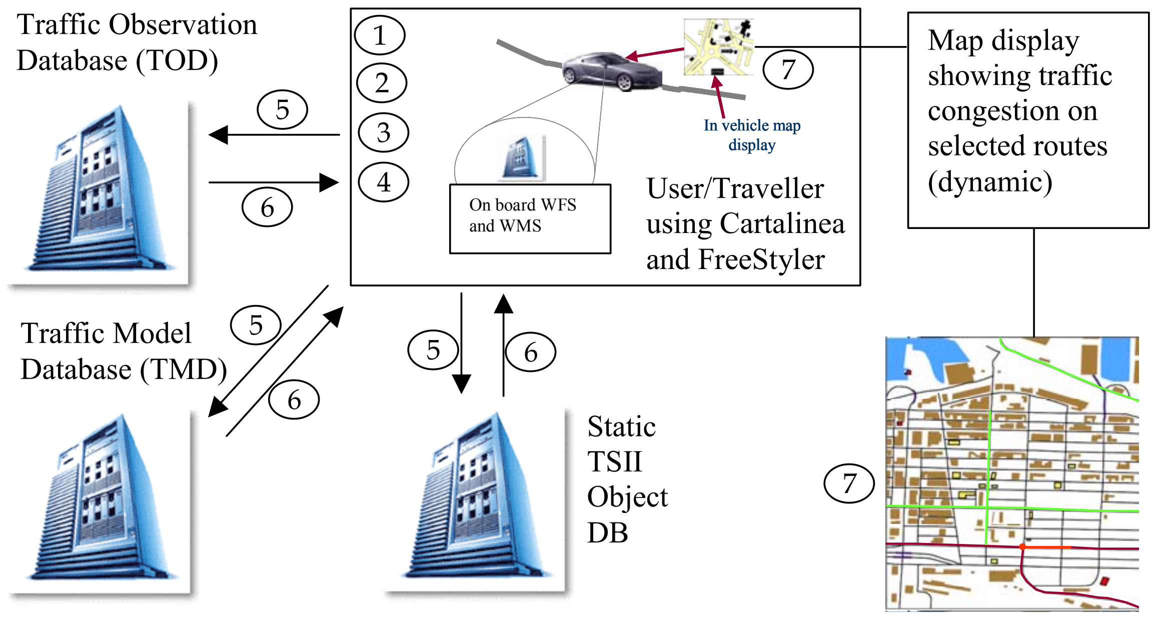

WFS Gateway
Traffic Observation Database (TOD)
Traffic Model Database (TMD)
Loop Detector Observation
1
2
3
4

2.2.8 User Requests a Map

Use case text description

Description	This use case describes how a user (e.g. vehicle traveller) with an onboard WMS/WFS requests a map with colour-coded traffic speed conditions overlaid.
Priority	High
Release	1
Actors	1. *User* 2. *FreeStyler WMS* 3. *Traffic Model Database (TMD) WFS* 4. *Road Network Server*
Pre-conditions	1. The *User* must be authenticated and authorized to access the *TMD*.
Events	1. The *User* selects the bounding box to view (e.g. coordinates of box containing his/her current location and intended route). 2. The *User* selects the Feature layer(s) to view (e.g. *Roads*, *Links*, *IncidentReports, LoopDetectors*), which can come from multiple distinct WFSs (e.g. *TMD, TOD,* and *Road Network Server*). 3. The *User* selects a traffic congestion style (colour-coded *Links*). 4. The *User* submits a Get Map request to *FreeStyler*. 5. *FreeStyler* submits Get Feature request to *TMD, TOD,* and *Road Network Server*. 6. *TMD, TOD,* and *Road Network Server* return features to *FreeStyler*. 7. *FreeStyler* displays map with overlaid colour-coded traffic speed conditions.
Alternate Paths	
Post-conditions	
Notes	The colour coding (e.g. a red to green scale where red represents 0 mean speed, green represents highway speed limit or greater) is determined by a user-defined traffic congestion style.

Use case diagram— User Requests a Map


Traffic Observation Database (TOD)
1
2
3
4
5
6
7
In vehicle map display
On board WFS and WMS
User/Traveller using Cartalinea and FreeStyler
Map display showing traffic congestion on selected routes (dynamic)
Traffic Model Database (TMD)
Static TSII Object DB

3 TSII COMPONENTS AND ARCHITECTURE

The TSII prototype is implemented using four Web Feature Servers, two Web Map Servers, one Web Registry Server at Galdos Systems and the WFS Gateway at the University of Toronto ITS Lab. Data and communication between these servers in this model are intended to represent the dynamic and distributed data handling of a real-world Intelligent Transportation System. Sections 3.1-3.3 provide a detailed description of the components of the TSII and their inter-relationships.

3.1 Overview

Figure 3.1-1 gives a high-level overview of the TSII prototype architecture. The coloured boxes represent system servers and the labelled arrows indicate the type of transaction executed between the servers. The description of each server/system is summarized in Table 3.1-1.

Table 3.1-1 Descriptions of System Servers

System/Server	Description
WFS Gateway	Converts traffic sensor, incident and CMS data to WFS transactions (Insert, Update) to the dynamic traffic model and incident WFSs
Static TSII Object DB	A WFS that stores and distributes static objects such as sensors and road network data representing the 400 series highways
Traffic Model DB	A WFS that stores and distributes traffic link data
Observation DB	A WFS that stores and distributes loop detector observations, incident reports and current CMS displays
CRS Registry	A WRS that stores and allows for the discovery of CRS data (2D, and 1D Linear Reference Systems) and transformations
FreeStyler WMS	A WMS that styles and displays GML data as a map
Listener	A component of a WFS that detects an incoming transaction and triggers an appropriate action (e.g. replicate or create new feature)

The description of each WFS transaction is summarized in Table 3.1-2.

Table 3.1-2 Descriptions of WFS Transactions

Transaction	Description
Insert Feature	Adds a new feature or features with unique ID to the WFS
Update Feature	Modifies an existing feature or features in the WFS
Get Feature	Retrieves a feature or features from the WFS. The features retrieved can be selected by feature type, feature ID, spatial location, or by arbitrary property/attribute values
Describe Feature	Retrieves the GML schema definition of the feature or features from the WFS
Delete Feature	Deletes an existing feature or features in the WFS

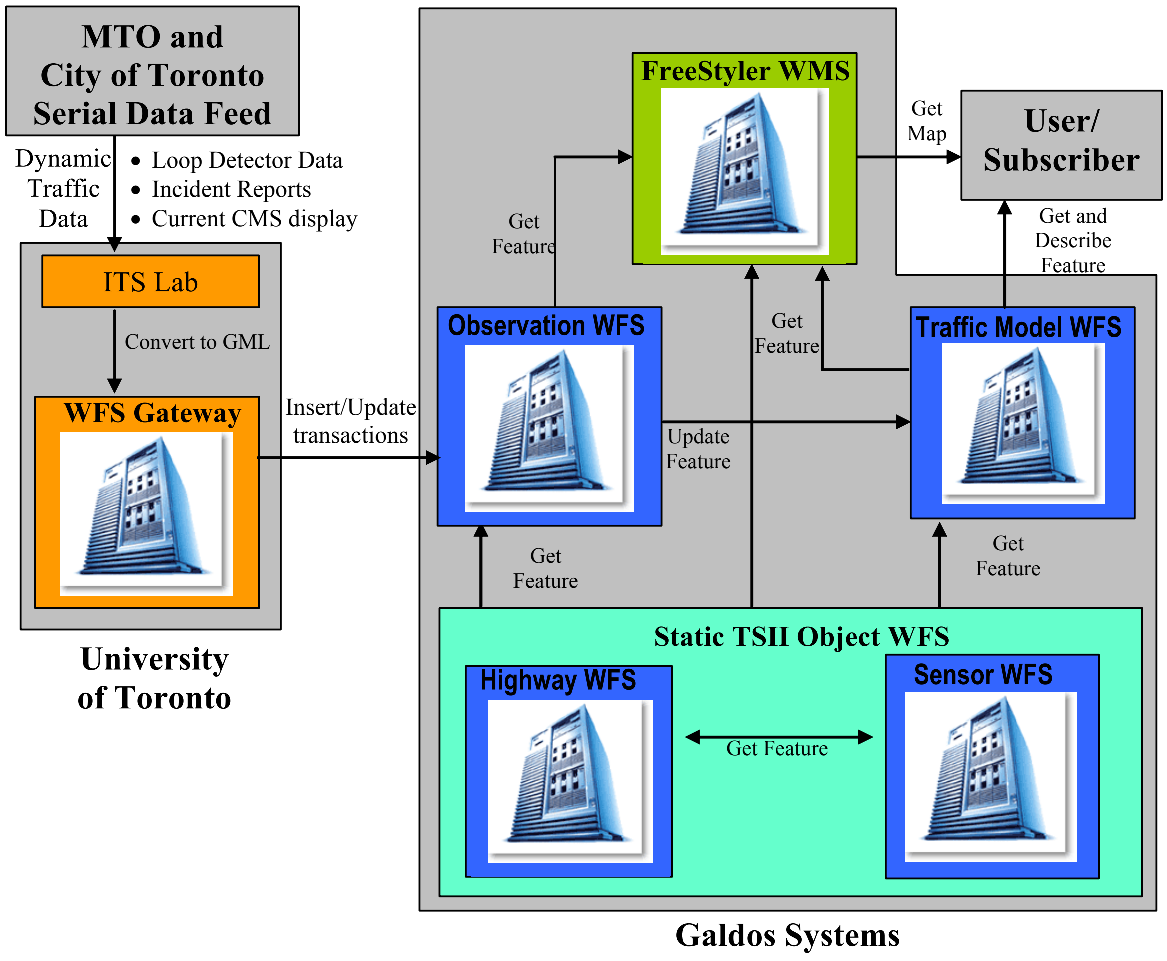


Figure 3.1-1 Overview of TSII Prototype Architecture

3.2 WFS Gateway

The Web Feature Service Gateway resides at the University of Toronto and consists of PHP scripts developed by the ITS Lab at the University of Toronto to convert the real-time MTO and City of Toronto data to GML Observations and then ouput to an XML file. Insert transaction requests are then created with the newly created XML file as payload and wrapped in the appropriate `<wfs:transaction>` tags. Java classes developed by Galdos Systems Inc. automatically submit the WFS Insert transaction over the Internet using the http post protocol.

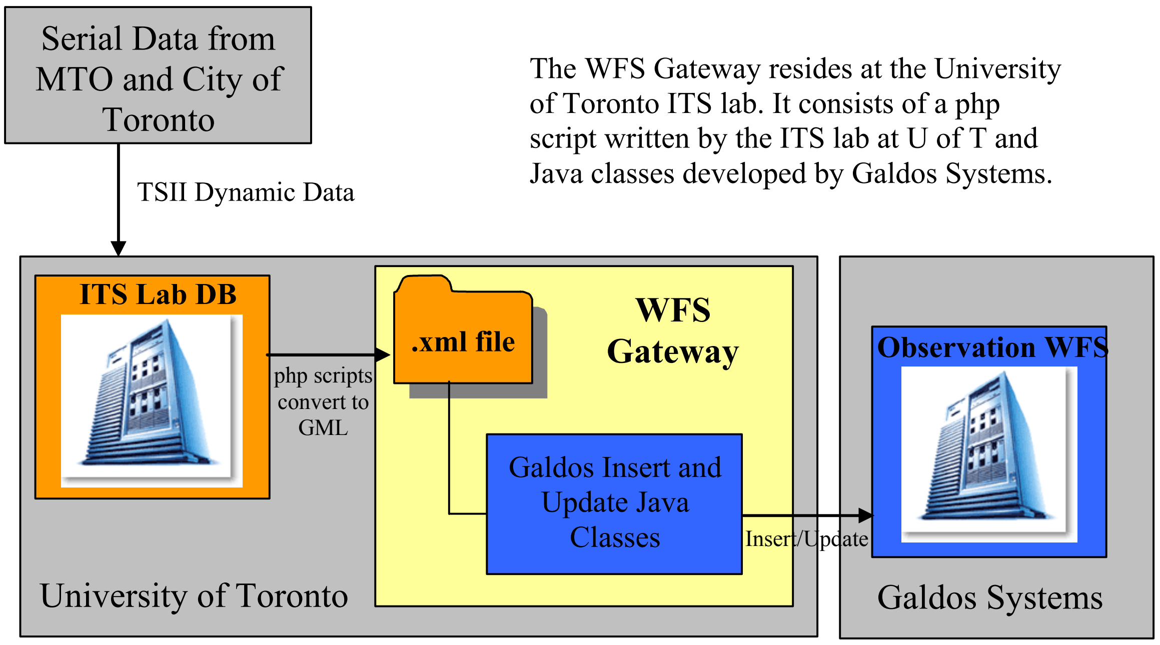

Serial Data from MTO and City of Toronto
TSII Dynamic Data
The WFS Gateway resides at the University of Toronto ITS lab. It consists of a php script written by the ITS lab at U of T and Java classes developed by Galdos Systems.
ITS Lab DB
php scripts convert to GML
.xml file
WFS Gateway
Galdos Insert and Update Java Classes
Insert/Update
Observation WFS
University of Toronto
Galdos Systems

Figure 3.2-1 WFS Gateway Architecture

3.3 Dynamic Traffic Data Handling

The traffic data is delivered to the ITS Lab at the University of Toronto from two serial ports at 19.2k baud, one for MTO and one for the City of Toronto. The physical interface is serial (RS-232)—a multi-kilometre "virtual" serial connection is provided by Siemens OTN fibre optic equipment. The incoming data format is binary, defined by some C structs and then serialized. An application then reads the data coming into the ITS Lab and stores it in a local MySQL database.

3.3.1 Loop Detector Readings

The most frequent source of dynamic traffic data received by the prototype TSII from MTO and the City of Toronto are loop detector readings. The loop detectors are located in each traffic lane roughly every half kilometre along the Toronto highways. The readings are taken every 20 seconds, measuring average speed, traffic volume and vehicle occupancy.

Sample loop detector data collected in a 20 second interval is shown in Figure 3.3-1.

Traffic Data from VDS 401DE0010DWC

Location: W OF KEELE CL

Upstream VDS: 401DE0020DWC --> Downstream VDS: 401DW0010DWC

Individual Loop data:
Lane: 1, US_Vol = 2, DS_Vol = 2, US_Occ = 1, DS_Occ = 1, Speed = 105
Lane: 2, US_Vol = 10, DS_Vol = 10, US_Occ = 10, DS_Occ = 10, Speed = 100
Lane: 3, US_Vol = 11, DS_Vol = 11, US_Occ = 11, DS_Occ = 16, Speed = 74

Overall averages for this VDS:Volume = 7.7 (veh/20sec), Occupancy = 8.2, Speed = 93.0 km/h

Data as of 2004-05-28 13:24:57

Graph of data available in the last hour:

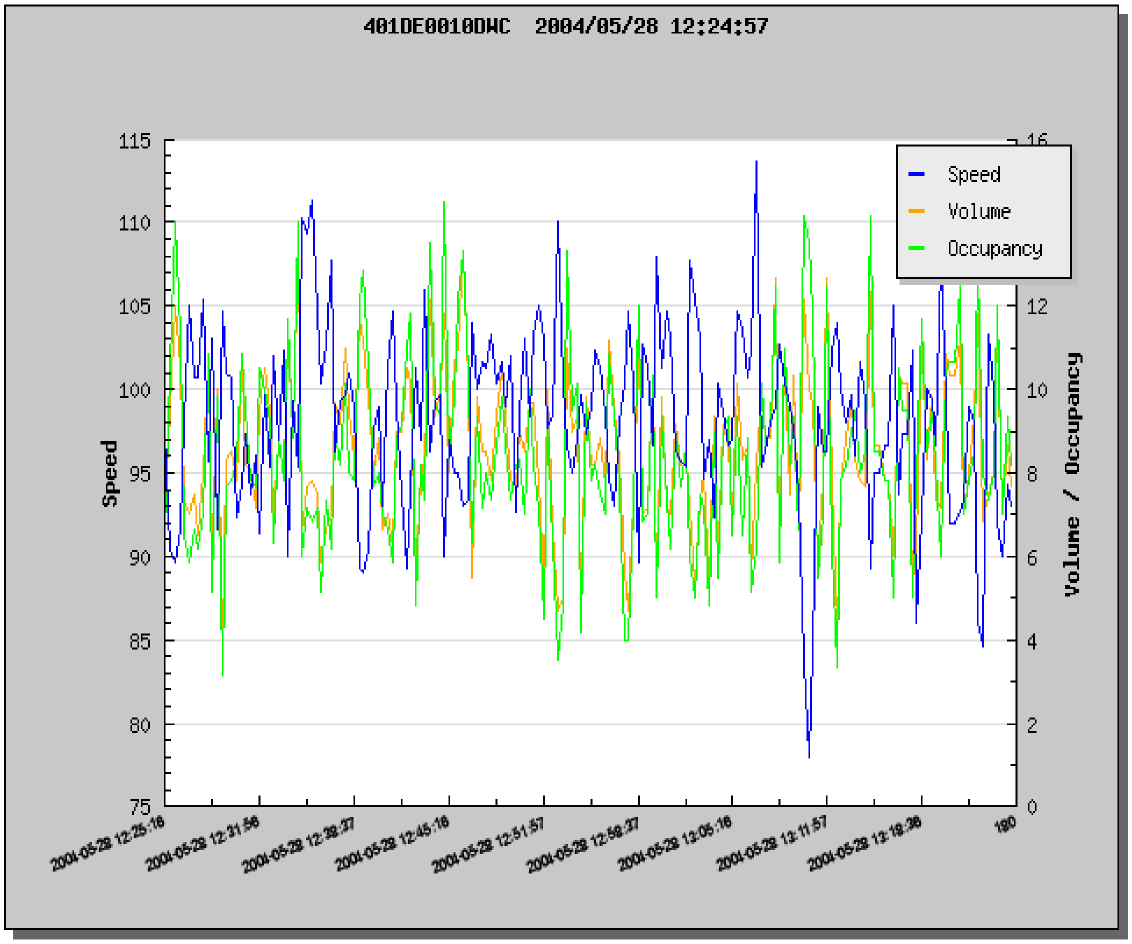


Figure 3.3-1 Sample Loop Detector Data Collected in 20 Second Intervals

The loop detector readings for each loop detector get converted to GML, which are encoded as a special type of Feature called a *LoopDetectorObservation* (see Appendix A for detailed definition). The properties values of the *LoopDetectorObservation* store the speed, traffic volume and vehicle occupancy readings. Other transportation features in the TSII that depend on these readings (e.g. the *Link* feature) are automatically created or updated as each *LoopDetectorObservation* is received by the Observation WFS. The *Link* feature represents a traffic link, which is a segment of the highway associated with a cluster of sensors (loop detectors) that cover all lanes in a single direction of traffic. Traffic links store the length, average speed along, and transit time of the highway segment, where average speed and transit time are derived from the loop detector observations. A "Listener" component of the Observation WFS monitors the transaction log and performs an Insert *Link* feature transaction to the Traffic Model WFS when a *LoopDetectorObservation* is inserted in the Observation WFS (see Section 2.2 for the technical details). The loop detector data flow is illustrated in Figure 3.3-2.

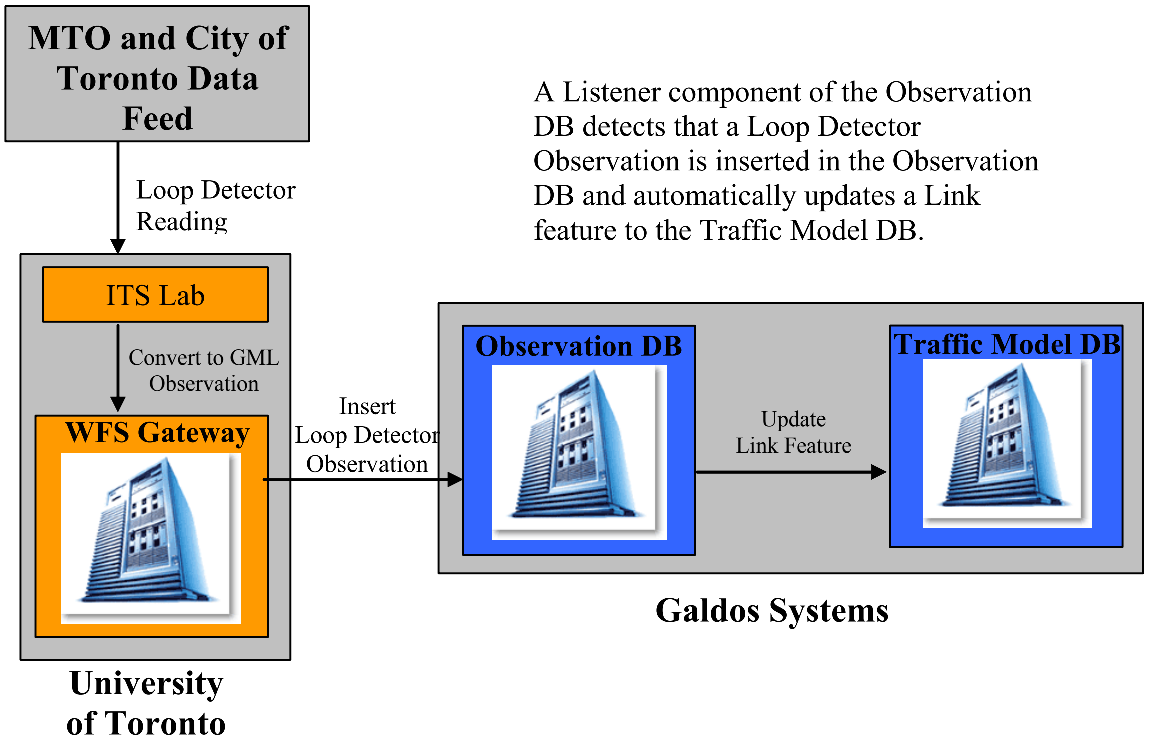

MTO and City of Toronto Data Feed
Loop Detector Reading
ITS Lab
Convert to GML Observation
WFS Gateway
University of Toronto
Insert Loop Detector Observation
A Listener component of the Observation DB detects that a Loop Detector Observation is inserted in the Observation DB and automatically updates a Link feature to the Traffic Model DB.
Observation DB
Update Link Feature
Traffic Model DB
Galdos Systems

Figure 3.3-2 Loop Detector Observation, Listener and Update Feature Action

3.3.2 Incident Reports

Incident reports are logged by human operators and conform to a specific file format. The City of Toronto and MTO each have different incident report formats and are both supported by the GML application schema for ITS. The City of Toronto incident report is structured as follows:

```
/* Used for static Event Data
*/
       short ateID,
       float latitude,
       float longitude,
       char* mainStreet,
       char* crossStreet

/* Used for Event Status update
 *
 *     Values for incidentCategory:
 *            ACCIDENT(1), CONGESTION(2), FIRE(3), DISABLED_VEH(4), POLICE(5),
 *             WEATHER(6), HAZARD(7), CLOSURE(8), OTHER(9), CONSTRUCTION(101),
 *            SPECIAL_EVENT(102)
 *
 */
       short ateID,
       char ateStatus, /* takes values OK(1), ERROR(0) */
       char* secondCrossStreet,
       char incidentType, /* takes values SCHEDULED(1), UNSCHEDULED(2), OTHER(3)*/
       char incidentStatus, /* takes values IN_PROGRESS(1), CLEARED(3), OTHER(4)*/
       char incidentCategory, /* see above */
       char incidentSeverity, /* takes values LOW(1), HIGH(3) */
       char* direction,
       char* lanesAffected,
       char days,  /* Duration */
       char hours, /* in BCD format */
       char minutes
```

The MTO incident report is structured as follows:

			INPUT	OUTPUT	
ACTIVE_INCIDENTS	**TYPE**	**SCALE**	**SCALE**	**DEC**	**PICTURE**
* INCIDENT_KEY	CHAR				X(26)
.DATE	DATE				YYYY/MM/DD
.TIME	NUM	0	0	0	^^^,^^^
.VDS_ID	CHAR				X(12)
* VDS_NUMBER	NUM	0	0	0	^^^
INCIDENT_STATUS	NUM	0	0	0	^
APID_DOWNSTREAM_VDS	CHAR				X(12)
OPERATOR_REDECLARE	CHAR				X(1)
ALGORITHM (3)	CHAR				X(1)
INC_AUTOMATIC_CMS_FLAG	CHAR				X(1)
CONTROLLER_WARNING	CHAR				X(1)
INC_DETECTION_CYCLE	NUM	0	0	0	^
END_OF_QUEUE_VDS_ID	CHAR				X(12)
CURRENT_EOQ_VDS	CHAR				X(12)
DATASET_TIME	NUM	0	0	0	^^:^^:^^
SOURCE	NUM	0	0	0	^^
CONFIRMED_INCIDENT	CHAR				X(7)
.CONFIRMED_TIME	NUM	0	0	0	^^:^^:^^
.FALSE_INCIDENT	CHAR				X(1)
.FTMS_USER	NUM	0	0	0	^^^
OPERATOR_INPUT	CHAR				X(54)
.INCIDENT_ZONE	CHAR				X(5)
.INCIDENT_TYPE	NUM	0	0	0	^
.STRATEGY	NUM	0	0	0	^^^
.BLOCKAGE_PATTERN	CHAR				X(10)
.INCIDENT_CONDITION	NUM	0	0	0	^^
.REC_RESPONSE	NUM	0	0	0	^^^^
.IMPL_RESPONSE	NUM	0	0	0	^^^^
.RESPONSE_CHANGE	CHAR				X(1)
.CURRENT_BLO_PATTERN	CHAR				X(10)
.BLOCKAGE_VDSID	CHAR				X(12)
.CURRENT_INC_CONDITION	NUM	0	0	0	^^
.CURRENT_STRATEGY	NUM	0	0	0	^,^^^
.CURRENT_IMPL_RESPONSE	NUM	0	0	0	^,^^^
UPDATE_TIME	NUM	0	0	0	^^:^^:^^
NUMBER_OF_UPDATES	NUM	0	0	0	^^
TERMINATION_TIME	NUM	0	0	0	^^:^^:^^
CLEARING_ALGORITHM	CHAR				X(1)
TERM_CONF_TIME	NUM	0	0	0	^^:^^:^^
TERMINATING_OPERATOR	NUM	0	0	0	^^^
COMMENT_AREA	CHAR				X(240)
.COMMENT (3)	CHAR				X(80)

The Observables.xsd schema defines an *IncidentReport* feature as an XML schema element with child property elements (some optional) that could represent the fields in either one of these two formats (see Appendix A for details.)

When an incident report is received by the ITS Lab, the WFS Gateway converts it to GML and performs an Insert/Update WFS transaction to the Traffic Observation Database (WFS) at Galdos Systems. Users of the TSII with access to a WFS can subscribe to receive traffic incident reports. In this case, each *IncidentReport* is automatically replicated to the user as soon as the *IncidentReport* is received by the Observation database. A “Listener” component of the Observation database monitors the transaction log and performs the same Insert *IncidentReport* feature transaction to the subscribing WFS when an *IncidentReport* is inserted in the Observation database (see Section 2.2 for the technical details). The incident report data flow is illustrated in Figure 3.3-3.

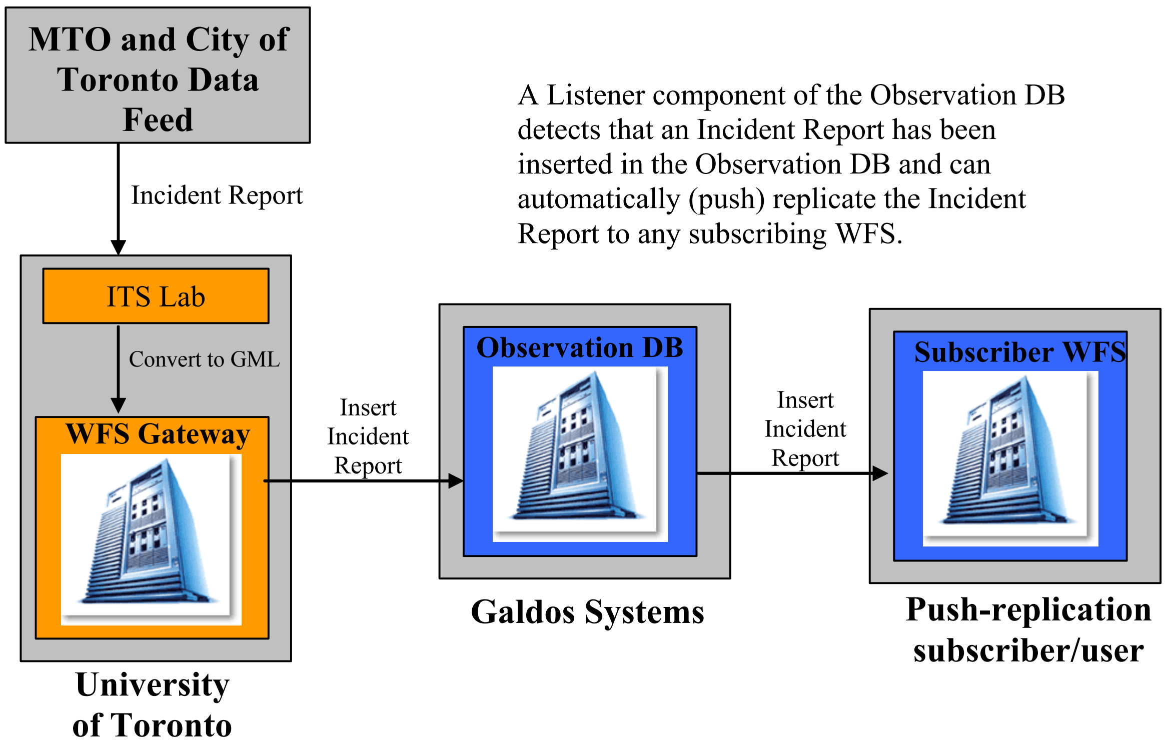

MTO and City of Toronto Data Feed
Incident Report
ITS Lab
Convert to GML
WFS Gateway
University of Toronto
Insert Incident Report
Observation DB
Galdos Systems
Insert Incident Report
Subscriber WFS
Push-replication subscriber/user
A Listener component of the Observation DB detects that an Incident Report has been inserted in the Observation DB and can automatically (push) replicate the Incident Report to any subscribing WFS.

Figure 3.3-3 Incident Report Listener and Insert Feature Action

3.3.3 Changeable Message Signs

The MTO and City of Toronto provide the current message display of 40 Changeable Message Signs (CMS) at various locations along the Toronto highways. When a CMS display is received by the ITS Lab, the WFS Gateway will convert it to a *MessageSignDisplay* feature in GML and perform an Insert/Update WFS transaction to the Traffic Observation database (WFS) at Galdos Systems. Users of the TSII with a WFS can receive the current *MessageSignDisplay* by retrieving it from the Observation database (WFS) (see Section 2.2 for the technical details) or also by push replication as in the case of incident reports (not depicted in Figure 3.3-3). The former case of the message sign display data flow is illustrated in Figure 3.3-4.

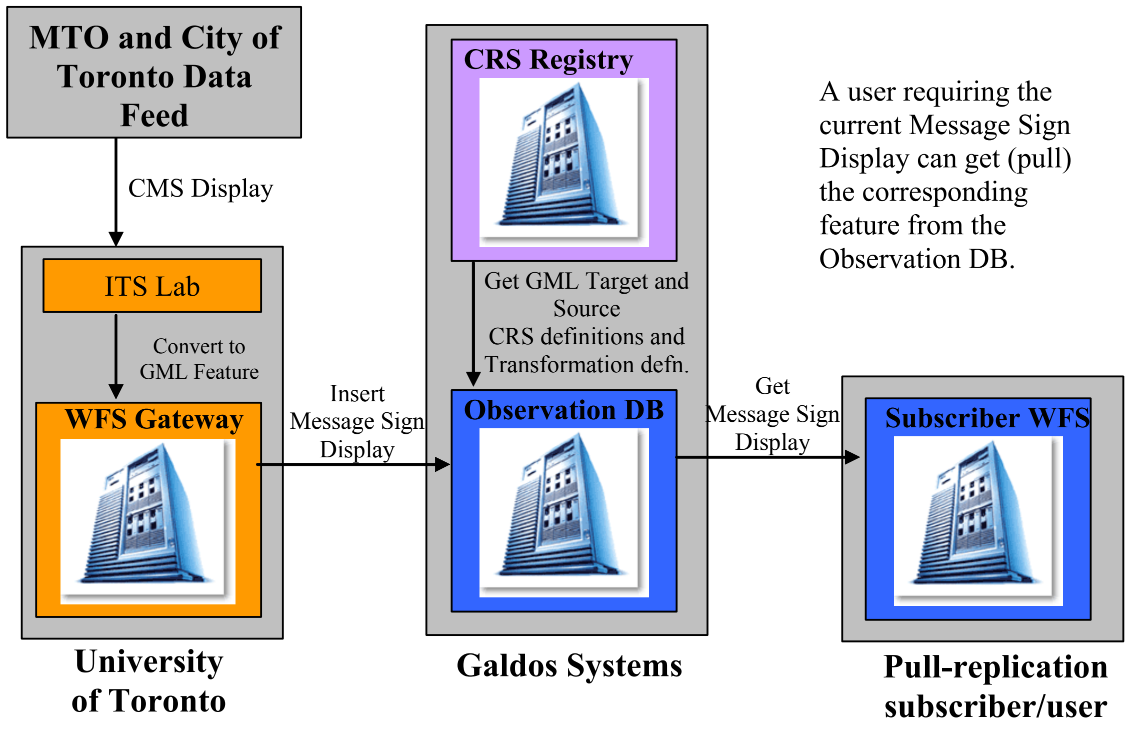

MTO and City of Toronto Data Feed
CMS Display
ITS Lab
Convert to GML Feature
WFS Gateway
University of Toronto
Insert Message Sign Display
CRS Registry
Get GML Target and Source CRS definitions and Transformation defn.
Observation DB
Galdos Systems
Get Message Sign Display
Subscriber WFS
Pull-replication subscriber/user
A user requiring the current Message Sign Display can get (pull) the corresponding feature from the Observation DB.

Figure 3.3-4 Insert MessageSignDisplay Transaction

3.4 TSII Schema Organization

The TSII application schema consists of four XSD files organized as follows, where solid arrows indicate inclusion of the target schema by the source and dashed arrows indicate import of the target schema by the source.

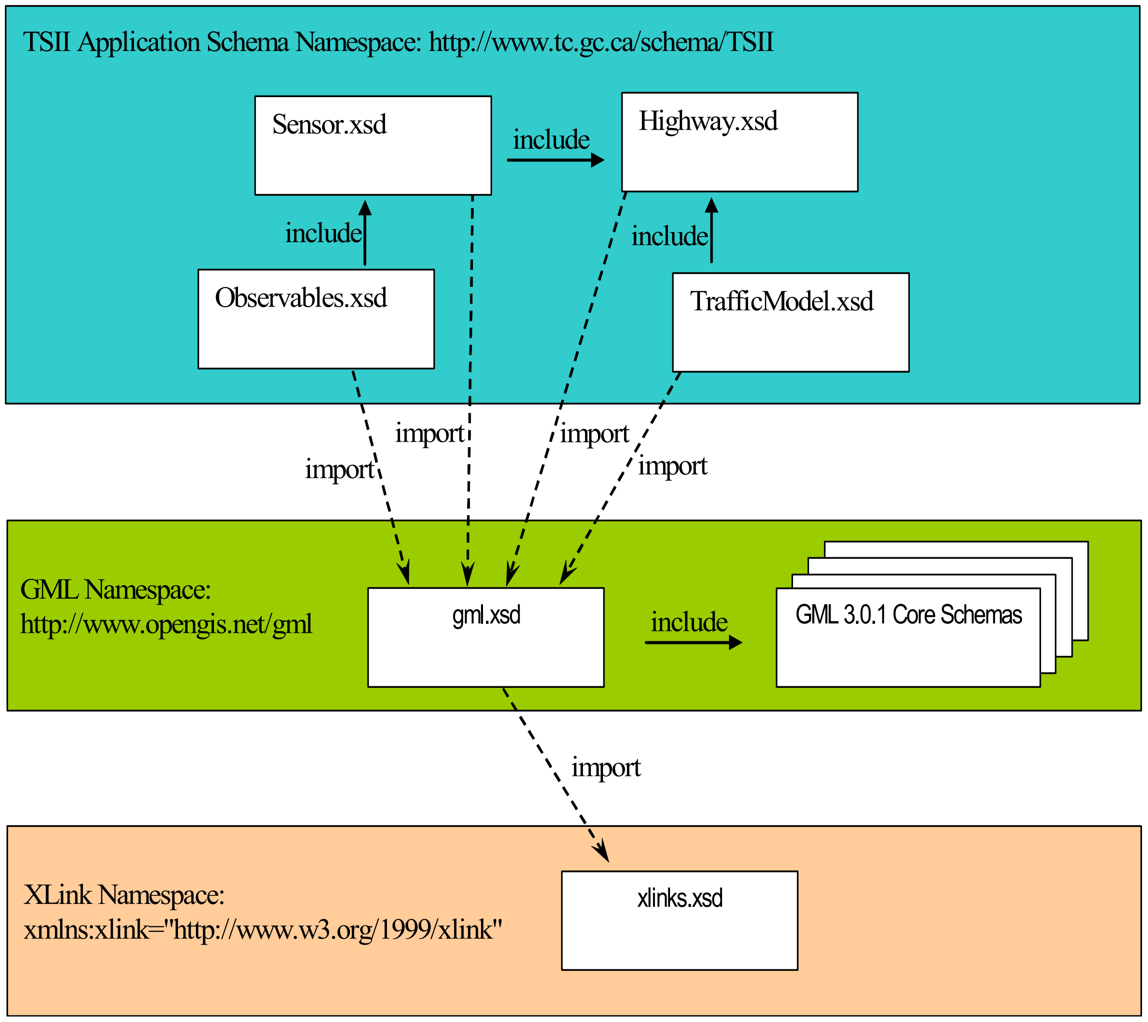


Figure 3.4-1 TSII Schema Organization

Each of the four schemas is stored in the WRS and in each of the corresponding Web Feature Servers: Traffic Observation WFS, Traffic Model WFS, Highway WFS, Sensor WFS. Appendix A lists the schema definitions.

4 PROCEDURE AND TESTS

4.1 Data Acquisition

Road and highway data was obtained from a data provider, NAVTEQ, in the ESRI Arcinfo format, which was then converted to GML version 2 using the FME Universal Translator. The resulting GML2 data was converted to GML version 3 using an XSLT stylesheet transformation. The stylesheet produced the final GML road and highway data that conforms to the TSII schema *Highway.xsd*. The sensor description data (location and identification) for MTO and City of Toronto Loop detectors was received from the University of Toronto ITS Lab and converted to GML using Perl scripts. This static sensor data conforms to the TSII schema *Sensor.xsd*. The traffic observables were received from the WFS Gateway and include loop detector observations, incident reports and changeable message sign dispays. The static sensor and highway data was loaded into the Highway WFS and the Sensor WFS, respectively. The observable data was loaded into the Traffic Observation WFS. The Traffic Model WFS was updated with dynamic traffic *Link* features automatically by the Listener/Action Manager.

4.2 Web Service Deployment

Each of the Web Feature Servers was deployed on a Galdos server with an underlying XHive (XML) database. The WMSs and WRS were also deployed on Galdos servers.

4.3 Software Extensions

The Listener component of Cartalinea was configured to "listen in" on WFS transactions (Insert, Update) and then check to see whether any existing policies apply to the type of transaction and its content, both before and after the transaction is persisted in the underlying database of the WFS. A policy was written in XACML (eXtensible Access Control Markup Language) that targets a successfully inserted *LoopDetectorObservation* and carries out the obligation of creating the Update *Link* transaction to the Traffic Model WFS.

4.4 Map Styling

The SVG map styling is encoded using OGC SLD (Styled Layer Descriptor). The colour-coding scheme of the *Link* features was adapted from that used by the MTO COMPASS Traffic Management System and is summarized in Table 4.4-1, where the column order is listed in order of priority. If the Average Speed is greater than zero, the colour indicated by the shown cell is used to style the *Link* symbol. If the average speed is 0, then the Traffic Volume column is used to determine the colour. If the Traffic Volume is zero, then the Sensor Occupancy (percentage of the 20 second interval the loop detector is triggered) column is used to determine the colour. In the case that no reading is reported from the sensor in the 20 second interval, the default values "-1" are set for Average Speed, Traffic Volume and Sensor Occupancy, and the corresponding colour code is grey.

Table 4.4-1 Colour-Coding Scheme for *Link* Features

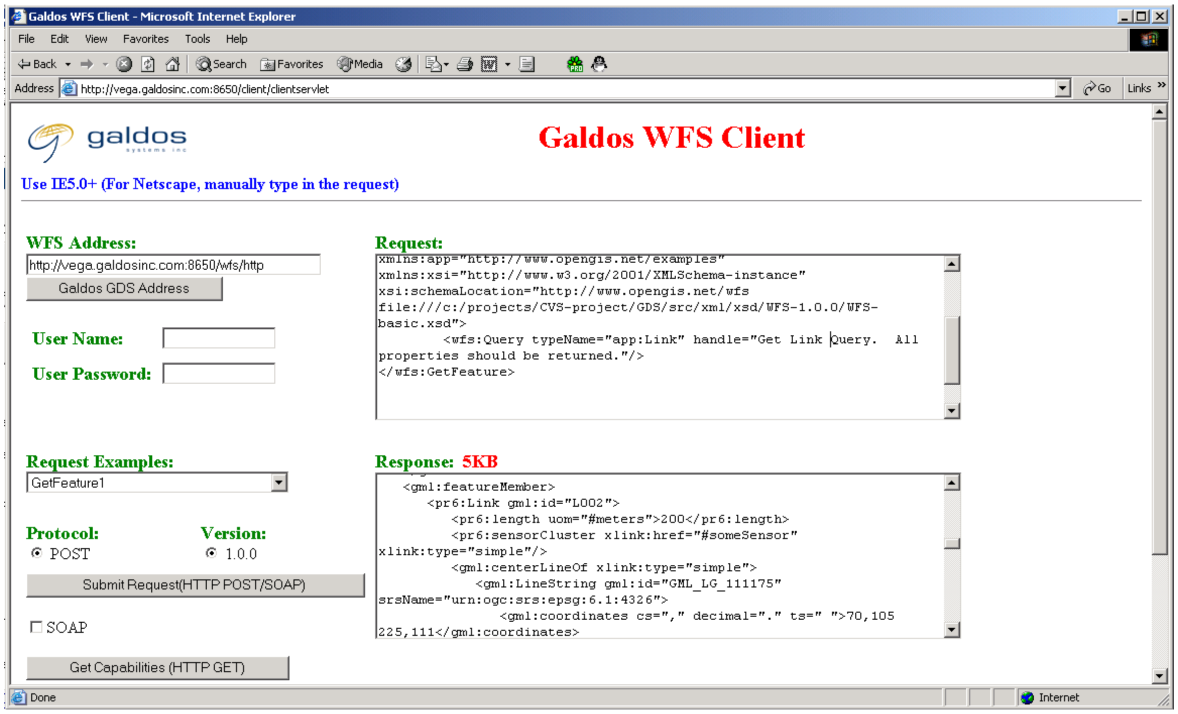
Average Speed (km/h)	Traffic Volume (vehicle count/20sec/lane)	Sensor Occupancy (%)
> 100	> 0 and < 4	0 - 6
75 - 100	4 - 8	6 - 14
50 - 75	8 - 12	14 - 22
25 - 50	12 - 16	22 - 32
> 0 and < 25	> 16	> 32
-1	-1	-1

4.5 Data Validation Tests

The Xerces J Version 2.6.2 Schema Parser was used to validate the road/highway instances, sensor instances, observation payload of the WFS Gateway insert transactions and the dynamic Link features.

4.6 WFS Transaction Tests

A WFS client developed by Galdos Systems was used to test the Insert, Get, Update, and Delete transactions on each WFS. A screen shot of the WFS client illustrating *Link* feature retrievel (Get transaction) from the Traffic Model WFS is shown in Figure 4.6-1.


Galdos WFS Client - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Go Links
galdos
Done
Internet

Address: http://vega.galdosinc.com:8650/client/clientservlet

Galdos WFS Client

Use IE5.0+ (For Netscape, manually type in the request)

WFS Address: http://vega.galdosinc.com:8650/wfs/http

Galdos GDS Address

User Name:

User Password:

Request Examples: GetFeature1

Protocol: [✓ POST]

Version: [✓ 1.0.0]

Submit Request(HTTP POST/SOAP)

☐ SOAP

Get Capabilities (HTTP GET)

Request:

```
xmlns:app="http://www.opengis.net/examples"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wfs
file:///c:/projects/CVS-project/GDS/src/xml/xsd/WFS-1.0.0/WFS-
basic.xsd">
        <wfs:Query typeName="app:Link" handle="Get Link Query.  All
properties should be returned."/>
</wfs:GetFeature>
```

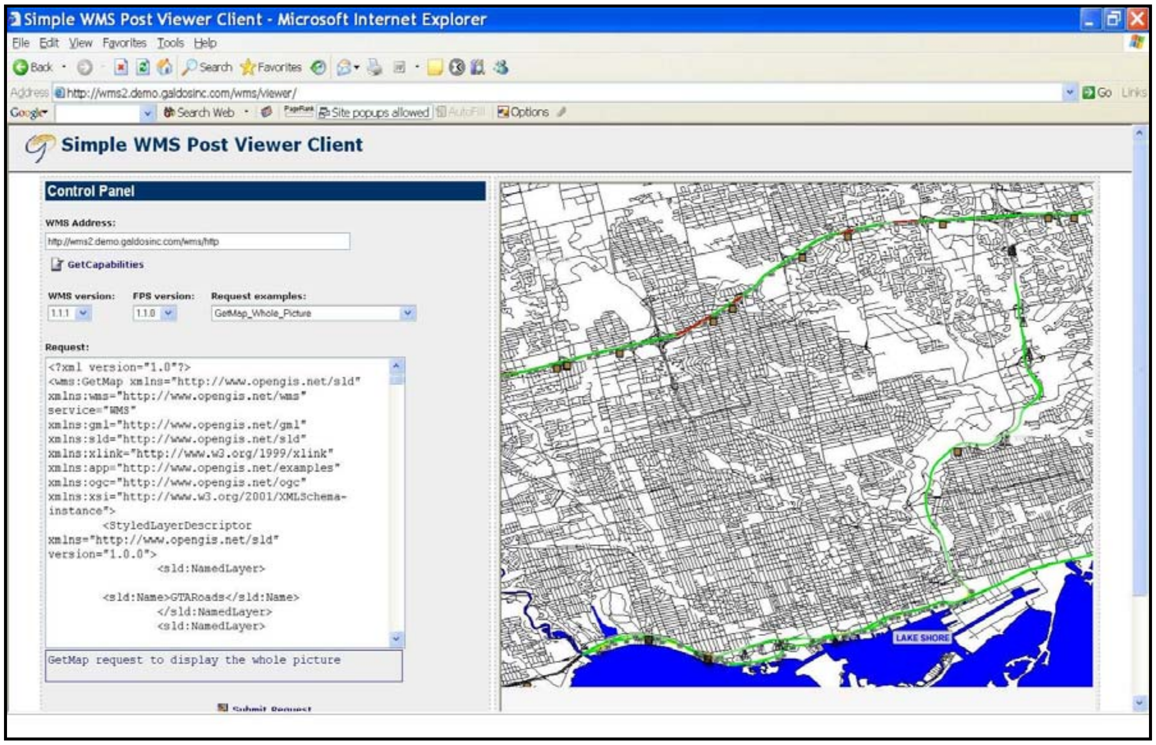
Response: 5KB

```
   <gml:featureMember>
      <pr6:Link gml:id="L002">
         <pr6:length uom="#meters">200</pr6:length>
         <pr6:sensorCluster xlink:href="#someSensor"
xlink:type="simple"/>
         <gml:centerLineOf xlink:type="simple">
            <gml:LineString gml:id="GML_LG_111175"
srsName="urn:ogc:srs:epsg:6.1:4326">
               <gml:coordinates cs="," decimal="." ts=" ">70,105
225,111</gml:coordinates>
```

Figure 4.6-1 Get *Link* Feature Transaction Results from Traffic Model WFS

4.7 WMS Transaction Tests

A WMS client developed by Galdos Systems was used to test the Get Map transactions. A screen shot of the WMS client illustrating feature portrayal in a standard browser of features (sensors, highways, message signs, incidents) from three of the TSII WFSs is shown in Figure 4.7-1. The TSII features are overlaid on a grid of non-highway (local) roads and Lake Ontario resulting from a Get Map request from the Galdos WMS client to an Intergraph WMS.


Simple WMS Post Viewer Client - Microsoft Internet Explorer
File Edit View Favorites Tools Help
http://wms2.demo.galdosinc.com/wms/viewer/
Simple WMS Post Viewer Client
Control Panel
GetCapabilities
Submit Request
LAKE SHORE

WMS Address: http://wms2.demo.galdosinc.com/wms/http

WMS version: 1.1.1 FPS version: 1.1.0 Request examples: GetMap_Whole_Picture

Request:

```
<?xml version="1.0"?>
<wms:GetMap xmlns="http://www.opengis.net/sld"
xmlns:wms="http://www.opengis.net/wms"
service="WMS"
xmlns:gml="http://www.opengis.net/gml"
xmlns:sld="http://www.opengis.net/sld"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:app="http://www.opengis.net/examples"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
        <StyledLayerDescriptor
xmlns="http://www.opengis.net/sld"
version="1.0.0">
                <sld:NamedLayer>

        <sld:Name>GTARoads</sld:Name>
                </sld:NamedLayer>
                <sld:NamedLayer>
```

GetMap request to display the whole picture

Figure 4.7-1 Get Map Result for Entire TSII Coverage from FreeStyler WMS

5 RESULTS

This project combined static road data and real-time loop detector data in a simple traffic scenario and implemented many of the basic components of the TSII using the four WFSs, two WMSs and the WFS Gateway. WFS software extensions were required and implemented to properly support the automatic update of dynamic features. In particular, the Listener/Action Manager component of Cartalinea (the Galdos WFS implementation) was developed in this project. The WFS Gateway produces *LoopDetectorObservations* encoded as GML from real-time MTO and City of Toronto sensor readings and supports *IncidentReports* and current *MessageSignDisplays*. The Galdos WMS client displays a road map requested from another OGC-compliant WMS from Intergraph and simultaneously portrays the TSII features including colour-coded traffic congestion overlaid on the road map in a standard web browser.

5.1 Portraying TSII features

The thin WMS client screenshot in Figure 5.1-1 portrays the TSII features as SVG overlaid on a road map from an Intergraph WMS to produce to a simple traffic flow diagram. When a mouse-over event is detected on a TSII feature, an HTML Tooltips text box is created to expose certain GML properties of the feature using JavaScript. The TSII *Road* features expose only the name (*gml:name* property value) when moused over, as shown in Figure 5.1-1.

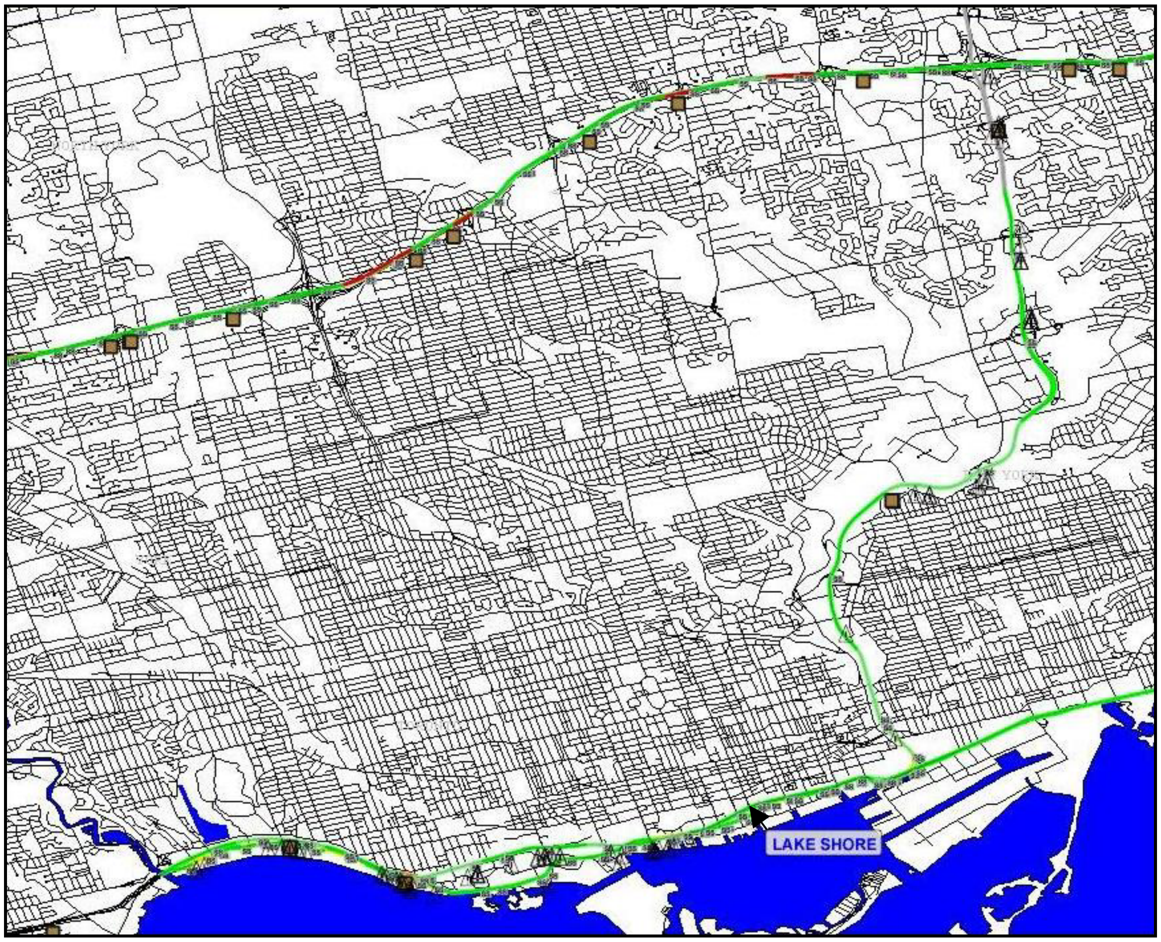
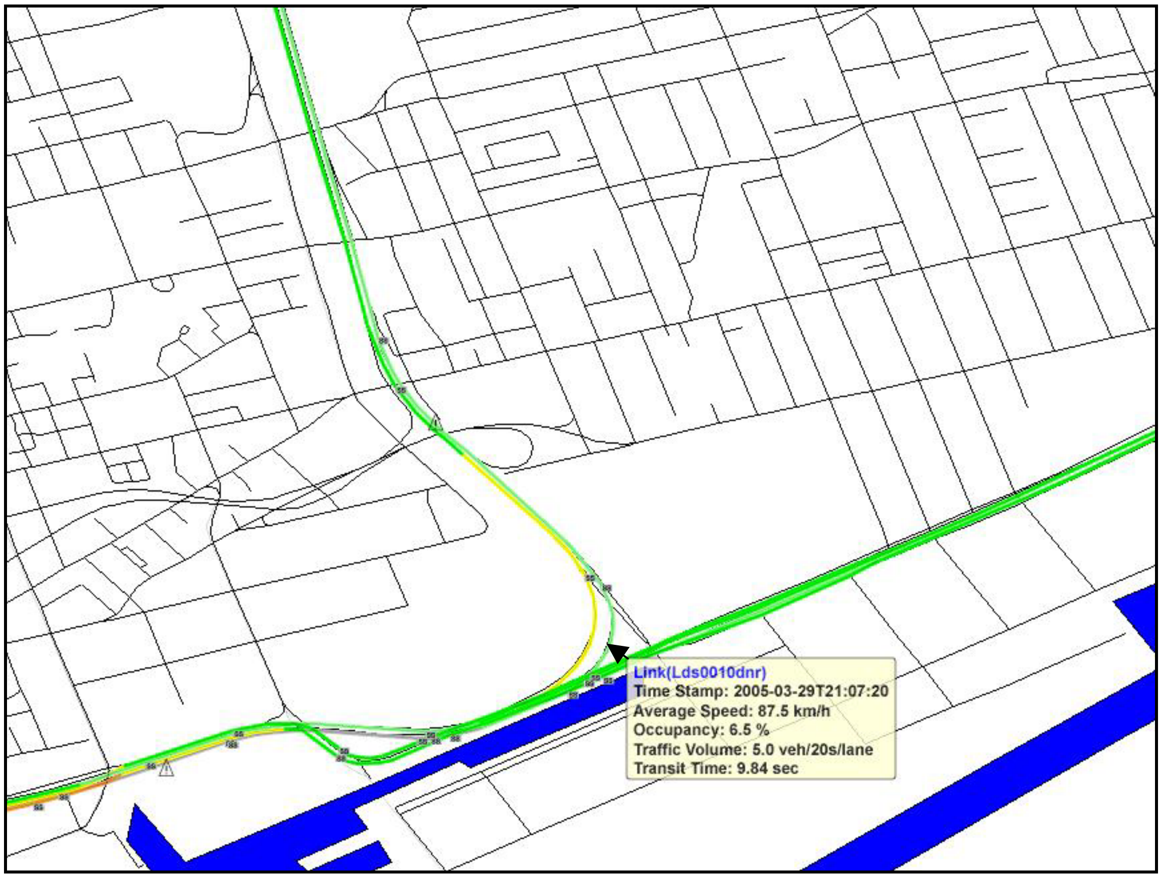


Figure 5.1-1 TSII Feature Portrayal in FreeStyler WMS

The traffic *Link* features are segments of the highway that are colour-coded from green to red according to Table 4.4-1 in the WMS client. When moused over, the Tooltips text box lists the values of the following properties and attributes of the Link feature:

- gml:id (unique identifier)
- timeStamp (XML Schema dateTime value)
- averageSpeed (in km/h)
- occupancy (%)
- trafficVolume (vehicle count/20 sec/lane)
- averageTransitTime (average time in seconds expected to traverse the *Link*)

Figure 5.1-2 illustrates the Tooltips mouse-over of a *Link* along Don Valley Parkway.



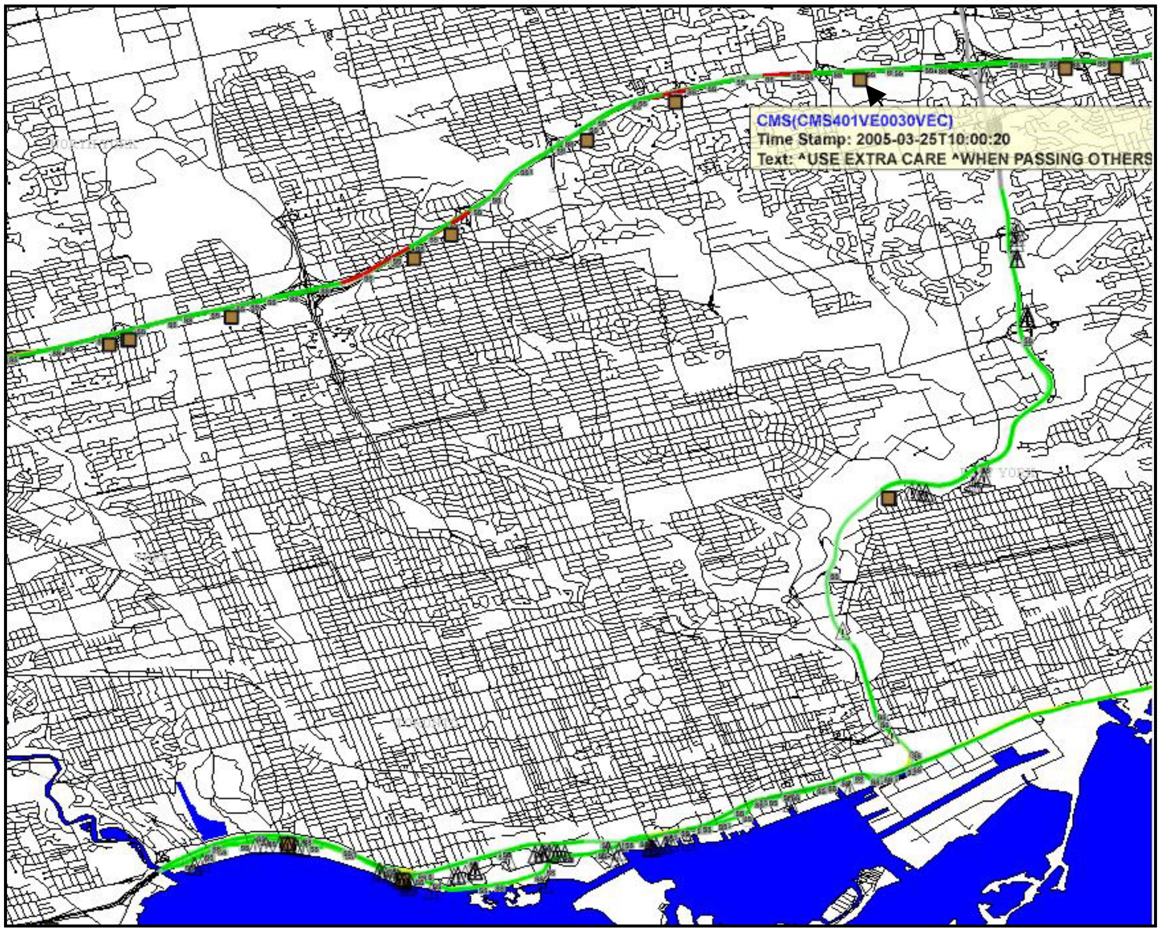
```
Link(Lds0010dnr)
Time Stamp: 2005-03-29T21:07:20
Average Speed: 87.5 km/h
Occupancy: 6.5 %
Traffic Volume: 5.0 veh/20s/lane
Transit Time: 9.84 sec
```

Figure 5.1-2 *Link* Feature Mouse-Over in FreeStyler WMS

The changeable message sign features are located along Highway 401, Don Valley Parkway, Gardener Expressway, and Lake Shore Drive, and are represented by brown square point symbols. When a message sign feature is moused over, the Tooltips text box lists the values of the following properties:

- timeStamp (XML Schema dateTime value)
- text (message display)

Figure 5.1-3 illustrates the Tooltips mouse-over of a *MessageSignDisplay* on Highway 401.



CMS(CMS401VE0030VEC)
Time Stamp: 2005-03-25T10:00:20
Text: ^USE EXTRA CARE ^WHEN PASSING OTHERS

Figure 5.1-3 *MessageSignDisplay* Feature Mouse-Over in FreeStyler WMS

The *Sensor* (loop detector) features are located roughly every half kilometre along Highway 401, Don Valley Parkway, Gardener Expressway, and Lake Shore Drive, and are represented by rectangular point symbols marked with "SS" (acronym for Sensor System). When a *Sensor* feature is moused over, the Tooltips pop-up box lists the values measured by the loop detector (volume, occupancy, and/or speed) and the unique id.

Figure 5.1-4 illustrates the Tooltips mouse-over of a *Sensor* feature along Highway 401.

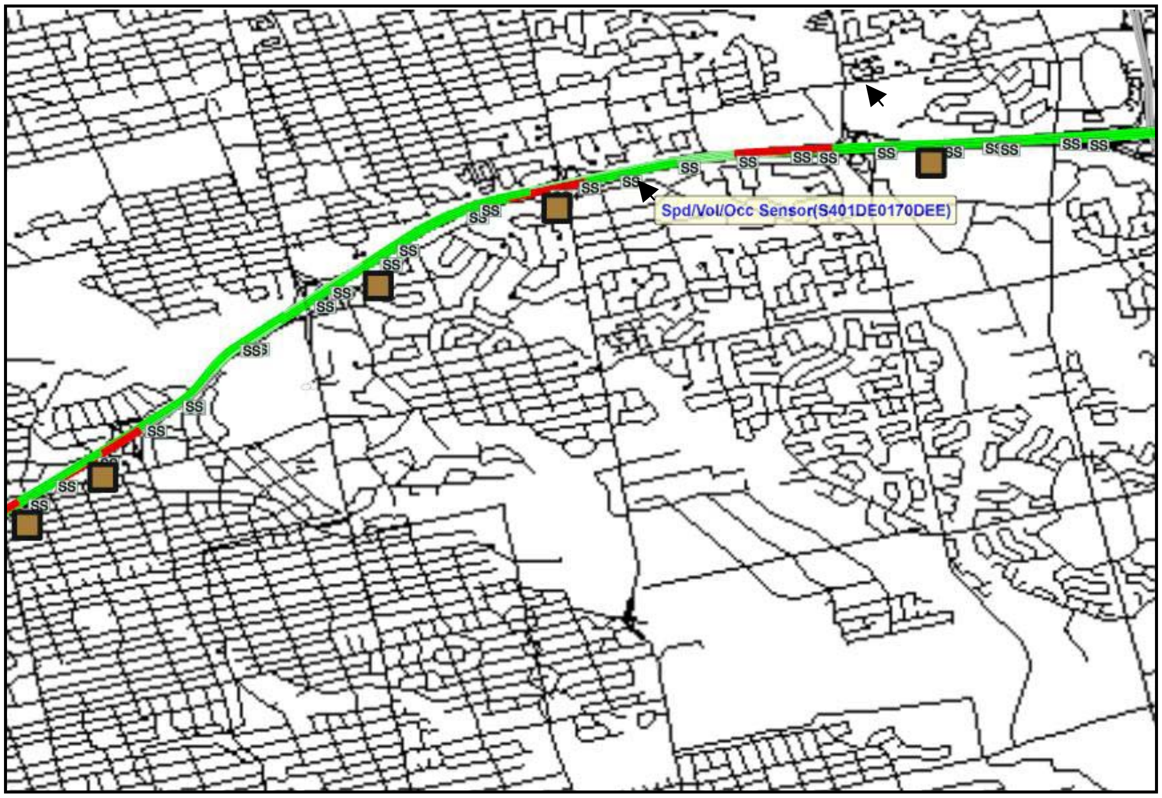


Figure 5.1-4 *Sensor* Feature Mouse-Over in FreeStyler WMS

The *IncidentReport* features are represented by triangle point symbols marked by an exclamation point "!". When a *IncidentReport* feature is moused over, the Tooltips pop-up text box lists the values of the following properties and attributes of the Link feature:

- timestamp (XML Schema dateTime)
- category (string)
- type ("Schedule" or "Unscheduled")
- condition (0 or 1, which corresponds to "in progress" or "cleared")

Figure 5.1-5 illustrates the Tooltips mouse-over of a *IncidentReport* feature on Lake Shore Drive.

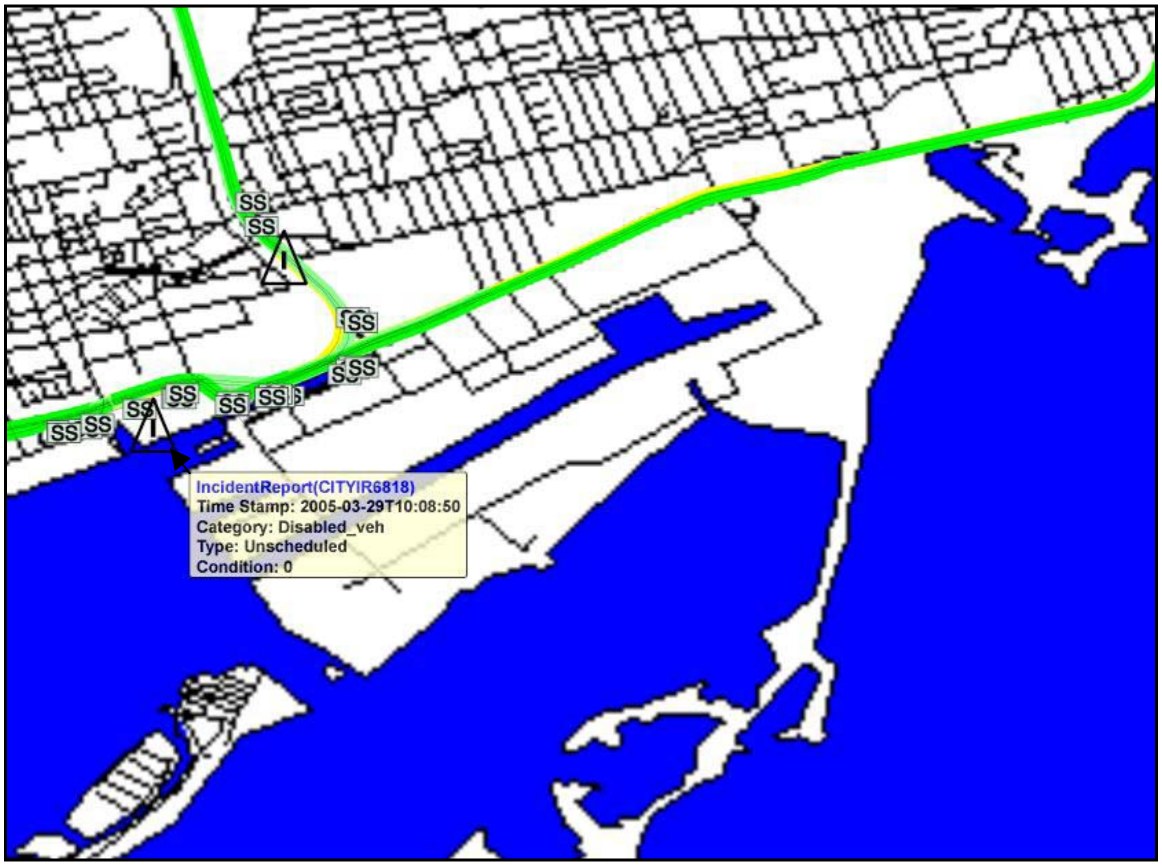


Figure 5.1-5 *IncidentReport* Feature Mouse-Over in FreeStyler WMS

6 ANALYSIS

The overall performance of the TSII prototype can be measured by the total delay from the time of the sensor reading to the instant the data is displayed on screen. Note that each server used in the TSII prototype is a Windows 2000 server that uses dual 2.8 GHz Xeon processors with 2GB DDR SDRAM. The complete data path is illustrated in Figure 6.1-1 and is decomposed as a sequence of events that are labeled on the diagram. The average duration for each event is measured and briefly described in Table 6.1-1.

6.1 Data Path

The data path in Figure 6.1-1 is decomposed into a sequence of seven events as shown.

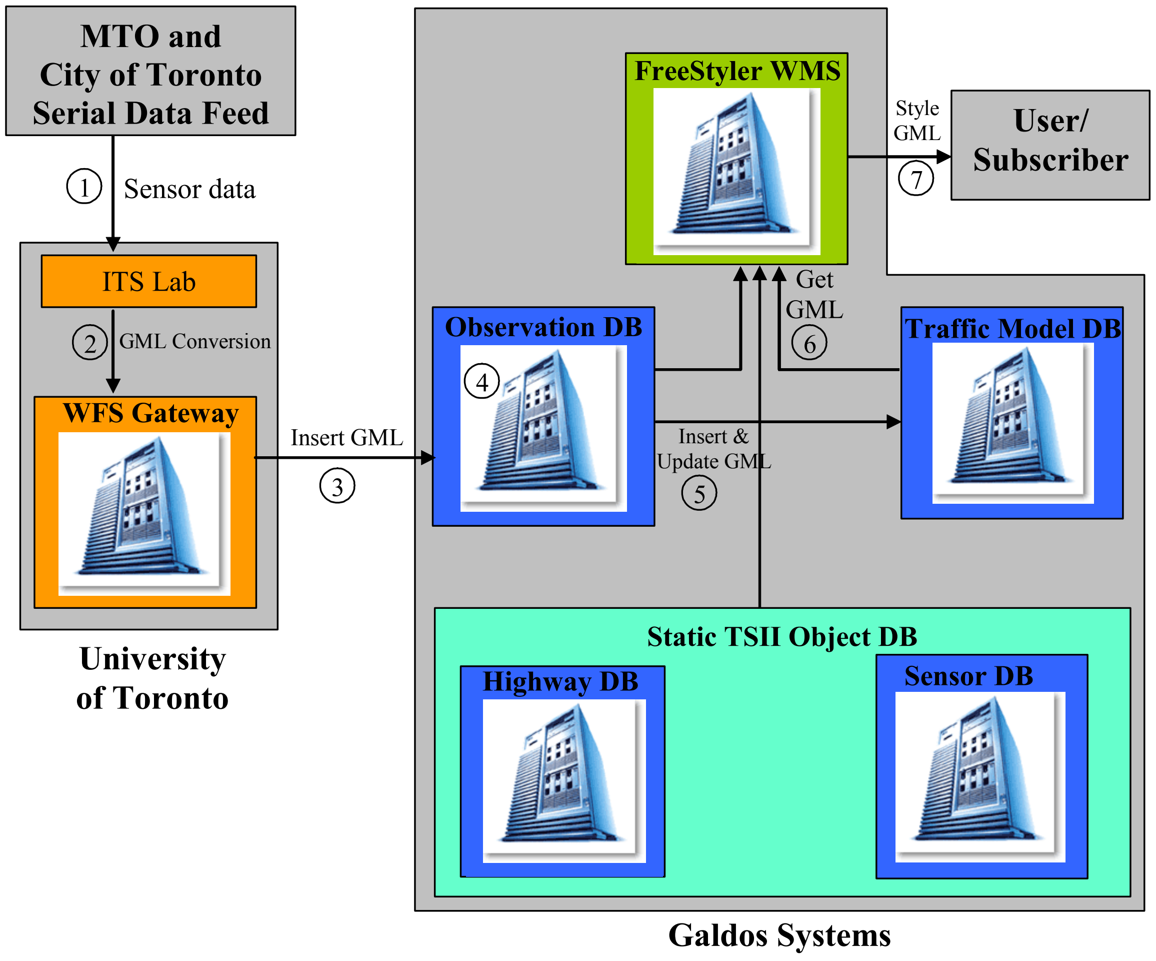


Figure 6.1-1 Data Path Decomposition

The first measured event is sensor data preparation and delivery to the ITS lab, which is broken down for the case of loop detectors as follows:

- A vehicle passes over a loop detector embedded under the road surface, creating a disturbance in electric current that is detected by the roadside Advanced Traffic Controller (ATC) device.
- Each ATC collects data (volume, occupancy and speed) over a 20-second interval for several clusters of loop detectors called Vehicle Detector Stations (VDS) in the area.
- A VAX server at MTO receives and stores the data from all ATCs on the highway.
- A process residing on the VAX server polls the database every second and listens for changes. Once a database update is detected, the current data is read and compressed (takes 2.5 seconds on average).
- The compressed data (10 KB on average) is sent through a 19.2 kbps serial link to the University of Toronto ITS lab (transit time is 11 seconds on average).

- The data received at the ITS lab is decompressed and then averaged over all lanes for each VDS before being inserted in a MySQL database (takes 0.9 seconds on average).
- A separate process polls the MTO database every 5 seconds, checking for updated CMS and Incident info. When an update is detected, the data is sent to the ITS lab immediately.

The total time delay corresponding to the sensor data preparation and delivery event is summarized in the first row of Table 6.1-1. The remaining six events in the data path sequence are described in Section 3 and the corresponding time durations are listed in Table 6.1-1. Where applicable, the primary protocol or processing technology used in each event is given in parentheses, e.g. (HTTP POST).

Table 6.1-1 TSII Prototype Performance Results

EVENT	DESCRIPTION	AVERAGE DURATION (seconds)	
		MTO data	City of Toronto data
1	Sensor data preparation and delivery to ITS lab database	20.0	18.00
2	GML conversion (PHP script)	0.1	0.09
3	Send request via WFS Gateway (HTTP POST) including network delay	1.6	1.10
4	Observation DB commitment (XHIVE Database)	171.0	110.50
5	Traffic DB update (Java based listener/action manager). This event is concurrent with event 4.	332.5	209.00
6	Get Feature request (HTTP GET)	5.5	5.00
7	Transform GML to SVG (XSLT)	2.1	2.00
	Total Time	**361.8**	**235.19**

7 CONCLUSION

The development of the prototype TSII based on GML and OGC web services shows that an open standard solution is capable of supporting dynamic traffic sensor information. Such an open standard solution has clear advantages, such as promoting interoperability, but the performance, although not unacceptable, could be improved. The performance results indicate an average lag time from sensor reading to display of roughly 6 minutes for MTO data and 4 minutes for City of Toronto data. There was insufficient time in this project for further investigation into reducing the lag time by optimizing software configuration, for example, which might have yielded better performance results.

The WMS client used in the TSII prototype retrieved and displayed SVG map data from two different web map servers, where the underlying feature data originated in different databases, with different storage formats. The WMS hosted by Galdos requested GML feature data from four WFSs with underlying XHIVE (native XML) databases and the WMS hosted by Intergraph retrieved data from an ArcSDE/Oracle database. The TSII implementation thus demonstrated the seamless integration, distribution, and display of transportation data from multiple sources with different underlying data storage formats, thus enabling wide area plug and play of transportation information. Other sources of static and dynamic data (weather sensors, GPS devices, etc.) in any existing database can be connected to the TSII via WFS and the resulting larger pool of transportation data can then be displayed in the same map or retrieved in the same result set for processing. The OGC open standards-based TSII thus enables wide area plug and play for transportation information.

REFERENCES

1. Institute of Transportation Engineers (ITE), *TMDD Center-to-Center Concept of Operations and Requirements Standard* (www.ite.org/tmdd/index.asp). June 2003.
2. ISO/DIS Draft International Standard (ISO/TC 204) *GDF-Geographic Data Files.* 2001.
3. ITS Architecture for Canada (www.itscanada.ca/english/architecture.htm), 2005.
4. Ministry of Transportation of Ontario (MTO), *ITS Internet Applications: Data Exchange Standards Report* (www.travelerinformation.com/itsproject/Enterprise_Aug1200/Data Exchange Report-Final.htm), November 2000.
5. Ministry of Transportation of Ontario (MTO), *ITS Internet Applications: Web Site Review Report* (www.travelerinformation.com/itsproject/Task1Report/ReviewReport.html), 2000.
6. National Cooperative Highway Research Program (NCHRP). *TransXML* (in progress). www.camsys.com/
7. Open Applications Group (OAGI) (www.openapplications.org/), *TranXML*. November 2002.
8. Open Tourism Consortium (OTC) (www.opentourism.org/wiki/wiki.phtml), *TourML*, 2003.
9. OpenGIS Consortium (OGC 03-006r3), *Open Location Services*. January 2004.
10. Prasad, B., *Role of XML in Transportation*. www.wipro.com/insights/xmltransportation.htm. 2001.
11. Putman, J.R., *Architecting with RM-ODP*. Prentice Hall, 2000.
12. Sabounghi, R.L., *Modelling the Impact of Intelligent Transportation Systems on Surface Goods Movement in Canada*, Ph.D. Thesis. July 1997.
13. Sabounghi & Associates, et al. *Study on ITS Applications within the Canadian Trucking Industry* (www.tc.gc.ca/pol/EN/Report/its/menu.htm). November 1999.