



LACMTA

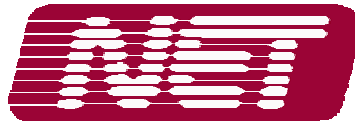
Regional Integration of Intelligent Transportation Systems (RIITS)

INTEGRATION INSTRUCTIONS

Version 1.0

December 2004

Prepared by:



National Engineering

Technology Corporation

14320 Firestone Blvd, Suite 100

La Mirada, CA 90638

REVISION TABLE

Date	Version	Description	Author
12/10/2004	R01	Release 1.0	NET Corp.



TABLE OF CONTENTS

1.0 INTRODUCTION	1-1
1.1 Purpose	1-1
1.2 Process Overview.....	1-1
1.3 Scope	1-1
1.4 Definitions, Acronyms, and Abbreviations	1-1
1.5 Document Overview	1-2
2.0 GETTING STARTED	2-1
2.1 XML Schemas	2-1
2.2 Sample XML Files	2-1
2.3 WSDL Files.....	2-1
2.4 About the Data.....	2-2
2.5 Sample Code.....	2-2
3.0 GENERATING WEB SERVICES LIST	3-1
3.1 Automatically Generating WSDL	3-1
3.2 Manually Generating WSDL	3-2
4.0 RIITS – CONNECTING WITH JAVA WEB SERVICES	4-1
4.1 Environment Setup	4-1
4.2 README File	4-2
4.3 Build SOAP Call	4-2
4.4 Modify Utility	4-2
4.5 Build & Run Sample Code.....	4-2
5.0 RIITS – CONNECTING WITH MICROSOFT .NET	5-1
5.1 Environment Setup	5-1
5.2 Update Web Reference	5-1
5.3 Modify Class1	5-3
5.4 Build & Run Sample Code.....	5-3
APPENDIX A: LIST OF ACRONYMS AND TERMS	A-1



LIST OF TABLES

TABLE 2-1: DATA TYPES AVAILABLE 2-2

LIST OF FIGURES

FIGURE 3-1: WEB SERVICES DEFINITION LANGUAGE 3-2
FIGURE 3-2: SAMPLE WSDL FILE 3-3
FIGURE 4-1: SAMPLE CODE DIRECTORIES 4-1
FIGURE 5-1: MICROSOFT .NET PROJECT 5-1
FIGURE 5-2: UPDATING WEB REFERENCES..... 5-2



1.0 INTRODUCTION

The Regional Integration of Intelligent Transportation Systems (RIITS) project involves multiple transportation modes and multiple transportation agencies within Los Angeles County in the formation of a regional network that supports transportation information exchange between agencies. The RIITS project utilizes a three-pronged approach to accomplish the goal of regional integration. First, the project creates an institutional process for managing the development of a regional network. Secondly, the project develops and implements a regional communications network to support inter-agency data exchange. Thirdly, it develops software tools that allow agencies and private companies to connect and integrate to the network.

1.1 Purpose

The purpose of this document is to provide technical instructions in which an agency or private company can connect to the Regional Network and extract data. This document assumes that the reader is a Software Engineer familiar with Java, XML, and Web Services Technology. For more information about XML, and Web Services please refer to the following web sites:

- <http://www.w3c.org>
- <http://webservices.org>
- <http://www.xml.org>

1.2 Process Overview

Accessing RIITS data involves multiple steps. The first step is to request access to the RIITS Network. This request can be made from the RIITS web site <http://www.riits.net>. When this request is processed by MTA, forms will be sent to the company. Some of the information requested will deal with the Agreements between agencies, companies and MTA. On one of the forms, there is a request for an IP address. This IP address should be the location from which all Programmatic Web Services requests will be sent. The IP address will be that of a Firewall in some configurations, or could be the IP address of the PC connecting to the Internet. Once all forms are processed by MTA, a username and password will be electronically sent to the agency or company. The instructions contained within this document will utilize this username and password.

1.3 Scope

The Scope of this document involves describing the sample code to access RIITS data, and instructions on modifying the sample code. This document is designed for a software engineer that is familiar with programming and running web services applications.

1.4 Definitions, Acronyms, and Abbreviations

Appendix A contains a glossary to provide a consistent set of definitions for the system.



1.5 Document Overview

The remainder of this document is organized into the following sections:

Section 2 – Provides a description of how to Get Started accessing RIITS data.

Section 3 – Describes how to generate WSDL Files.

Section 4 – Describes how to integrate to RIITS when using Java Web Services.

Section 5 – Describes how to integrate to RIITS when using Microsoft .NET.



2.0 GETTING STARTED

After a username and password have been received, technical staff should become acquainted with RIITS documentation. The documentation that should be downloaded from the web site includes:

- XML Schemas
- Sample XML Files
- WSDL Files
- Sample Test Code

2.1 XML Schemas

The XML Schemas describe the data available from the RIITS network, and also contain the format and data-types of the XML. These XML schemas can be compiled and source code generated, if desired, using web services tools. XML Schemas are available for Inventory and Real-time data. Inventory data is static configuration information. This information represents field device locations and does not generally change frequently. Inventory data includes device ID, Device Name, and Device Location information such as latitude and longitude of the field device. Inventory information is updated from agencies once a day at midnight. For this reason, it is not necessary nor recommended that Inventory data be retrieved more than once a day. Real-time data includes regularly updated data such as speed, volume, occupancy, and status. The request and response formats are also included. These describe how to format an XML data request and the format including header information for the XML documents received in response.

XML Schemas have been generated using the ATIS SAE ITS Standard. The ITS Standards allow for extensions to include data or data types not accounted for. RIITS has extended the standards to include XML Schemas for all data available.

2.2 Sample XML Files

Sample XML files are also available on the RIITS web site. (<http://www.riits.net>) The sample XML files for each data type available are actual files that were generated and sent from the RIITS system. These files can be used for testing purposes or for XML and XML Schema validation.

2.3 WSDL Files

Web Services Definition Language (WSDL) files are also available on the RIITS web site. The sample WSDL files exist for each data type. In subsequent sections of this document, the user will be shown how to dynamically generate WSDL files. The files contained on the Internet are sample WSDL files.



2.4 About the Data

As noted in above sections, many data types are available from RIITS. This data is updated at different rates. **It is imperative that users requesting data from RIITS do not utilize bandwidth by requesting data more frequently than it is updated.** The following table is a summary of the data that is available and the update rates in which this data is refreshed.

Table 2-1: Data Types Available

Data Type	Agency Providing Data	Update Rates
Congestion - Freeway Inventory	Caltrans D7	Daily/Midnight
Congestion – Freeway Real-time	Caltrans D7	Every 1 Minute
Congestion – Arterial Inventory	LADOT	Daily/Midnight
Congestion – Arterial Real-time	LADOT	Every 1 Minute
Congestion – Arterial Inventory	Caltrans D7	Quarterly
Congestion – Arterial Real-time	Caltrans D7	Every 1 Minute
Event	CHP	Every 1 Minute
Event	Caltrans D7	Every 1 Minute
Bus Routes – Inventory	MTA – Metro	Quarterly
Bus Real-time Locations	MTA – Metro	Once per 2 Minutes
Train Routes – Inventory	MTA – Metro	Quarterly
Train Real-time Locations	MTA - Metro	Every 1 Minute
CMS Inventory	Caltrans D7	Daily/Midnight
CMS Real-time	Caltrans D7	Every 1 Minute
CCTV Inventory	Caltrans D7	Quarterly
CCTV Snapshots	Caltrans D7	Every 1 Minute

2.5 Sample Code

Technical staff should then download the sample test code from the RIITS web site. Two types of sample code are available. One sample uses Java Web Services Technology and the other uses Microsoft .NET technology. The instructions in this document explain how to modify this sample code to connect to RIITS to receive data.

As mentioned above, RIITS data is available through Web Services. The most common technology used to transport Web Services, is the Simple Object Access Protocol (SOAP). As mentioned in the above section, this document assumes that the engineer using the Instruction Manual has an understanding of Web Services and understands components of Web Services such as SOAP.



3.0 GENERATING WEB SERVICES LIST

Web Services may be accessed directly using a SOAP call. The main steps to accessing a Web Service are:

- Build a SOAP Call
- Invoke the Service
- Parse the XML Response
- Extract the Required information

The remaining sections of this document will assist an engineer with Building the calls, and invoking the service. Parsing the XML and extracting “relevant” information is unique to each specific user.

3.1 Automatically Generating WSDL

To access the Web Services list and dynamically get the WSDL, open a web browser and go to the following URL: <http://rtsisp01.riits.net/ISP/services>.

At a minimum, the Congestion Web Service will appear. Please see Figure 3-1 for an example of the Web Services information available. To view WSDL using the browser, simply click on the <wsdl/> link. Every Web Service has a “get” method available. This “get” method is the mechanism to invoke RIITS Web Services.

To download the WSDL, right click on the <wsdl/> Link and select “Save Target As” option. Enter the file name, and the WSDL will be saved to hard disk. **NOTE: the name of the WSDL file MUST be the reserved name.** This file will be copied to the necessary directory in Section 4.0.

Reserved Names:

ISP_CongestionService.wsdl

ISP_CmsService.wsdl

ISP_CctvService.wsdl

ISP_BusService.wsdl

ISP_RailService.wsdl

ISP_EventService.wsdl



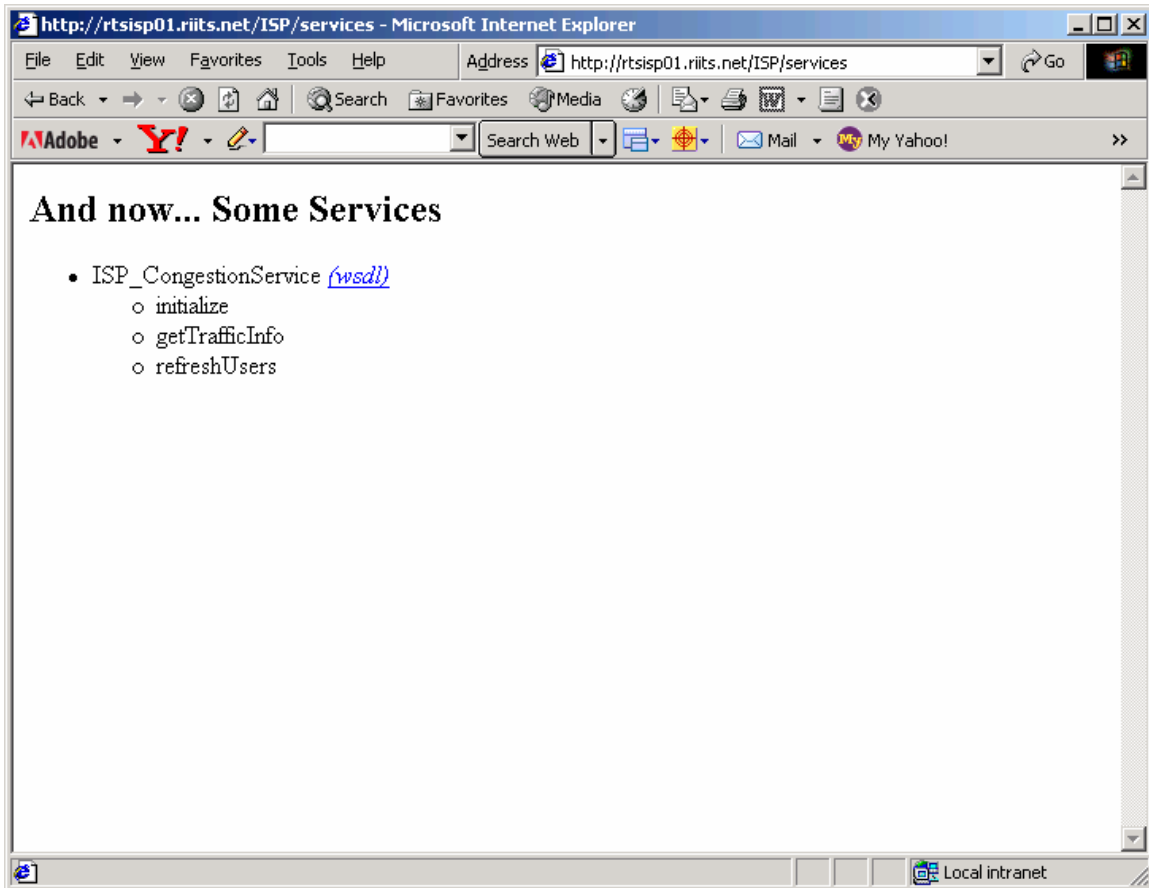


Figure 3-1: Web Services Definition Language

3.2 Manually Generating WSDL

To manually update the WSDL, copy the WSDL from the RIITS Internet site to a local area. Edit the WSDL file and modify the Namespace to the following IP address <http://www.rtsisp01.riits.net/ISP/services>.

Save the modified file for use in the subsequent section. Figure 3-2 is a sample WSDL file where the Namespaces can be seen.

NOTE: When saving WSDL files please note that the name of the file must be the reserved name used for the Web Service. This file will be copied to the necessary directory in Section 4.0.

Reserved Names:

ISP_CongestionService.wsdl

ISP_CmsService.wsdl

ISP_CctvService.wsdl

ISP_BusService.wsdl



ISP_RailService.wsdl

ISP_EventService.wsdl

```

<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions
  targetNamespace="http://149.136.124.21:9186/ISP/services/ISP_RailSe
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:apachesoap="http://xml.apache.org/xml-soap"
  xmlns:impl="http://149.136.124.21:9186/ISP/services/ISP_RailService"
  xmlns:intf="http://149.136.124.21:9186/ISP/services/ISP_RailService"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <wsdl:message name="getRailInfoResponse">
  <wsdl:part name="getRailInfoReturn" type="xsd:string" />
</wsdl:message>
- <wsdl:message name="initializeRequest">
  <wsdl:part name="agencyName" type="xsd:string" />
  <wsdl:part name="serviceName" type="xsd:string" />
  <wsdl:part name="configFile" type="xsd:string" />
</wsdl:message>
- <wsdl:message name="getRailInfoRequest">
  <wsdl:part name="userName" type="xsd:string" />
  <wsdl:part name="userPass" type="xsd:string" />
  <wsdl:part name="xmlMessageRequest" type="xsd:string" />
</wsdl:message>
<wsdl:message name="initializeResponse" />
wsdl:message name="refreshUserRequest"

```

Figure 3-2: Sample WSDL File



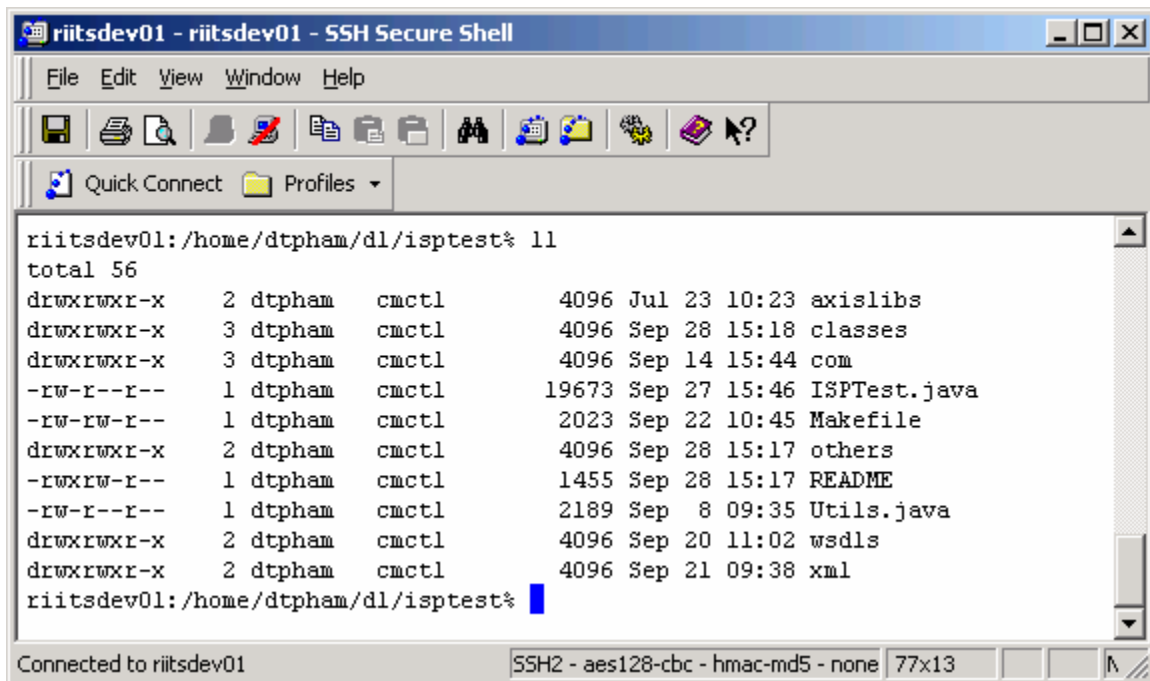
4.0 RIITS – CONNECTING WITH JAVA WEB SERVICES

This section describes how to use/modify the sample code to access RIITS data if using Java Web Services.

4.1 Environment Setup

If using Java Web Services, download the appropriate sample code. After downloading the appropriate sample code, unzip or untar the file. Figure 4-1 is a sample of the directories that should exist. Please note that the sample code was generated with the following criteria:

- Java SDK 1.4.2_03
- RedHat 9.0 Linux environment
- Java Axis 1.1 available at <http://xml.apache.org/axis/index.html>



```

riitsdev01:/home/dtpham/dl/isptest% ll
total 56
drwxrwxr-x   2 dtpham  cmctl      4096 Jul 23 10:23 axislibs
drwxrwxr-x   3 dtpham  cmctl      4096 Sep 28 15:18 classes
drwxrwxr-x   3 dtpham  cmctl      4096 Sep 14 15:44 com
-rw-r--r--   1 dtpham  cmctl    19673 Sep 27 15:46 ISPTTest.java
-rw-rw-r--   1 dtpham  cmctl     2023 Sep 22 10:45 Makefile
drwxrwxr-x   2 dtpham  cmctl      4096 Sep 28 15:17 others
-rwxrwr-r--   1 dtpham  cmctl     1455 Sep 28 15:17 README
-rw-r--r--   1 dtpham  cmctl     2189 Sep  8 09:35 Utils.java
drwxrwxr-x   2 dtpham  cmctl      4096 Sep 20 11:02 wsdl
drwxrwxr-x   2 dtpham  cmctl      4096 Sep 21 09:38 xml
riitsdev01:/home/dtpham/dl/isptest%

```

Figure 4-1: Sample Code Directories

Description of Sample Code:

- **Axislibs** – directory containing all libraries needed to support Axis.
- **Classes** – directory containing all compiled Java classes
- **Com** – directory containing all Java code for building communication classes
- **Java** – directory containing Java SDK 1.4.2_03
- **ISPTTest.java** – Java source code for accessing RIITS to get Congestion Data
- **Makefile** – Unix Makefile for building/compiling java code



- **README** – Step-by-Step instructions on what to modify before compiling.
- **Utils.java** – Java source that must be modified to contain the agency name, user name, and password sent by the RIITS Administrator.
- **WSDLs** – directory that contains all WSDL code. In the above section, WSDL was generated dynamically. This WSDL file(s) should be copied to this area.
- **XML** – directory used by sample code to write out retrieved XML files.

4.2 README File

The README File in the base directory describes step-by-step instructions for modifying the appropriate files for “Building/Compiling” the sample code. Follow these instructions.

4.3 Build SOAP Call

To build SOAP calls, begin with the WSDL files generated in the above section. Modify the Makefile to include the correct JAVAHOME (the location where Java is installed locally), and the IP address for the Web Service (URL from WSDL). To generate java classes from these WSDL file type:

```
make wsdl2j
```

This command should not create new directories. If a new directory was created make sure that the IP address in the makefile is the same as the IP address from the WSDL file.

4.4 Modify Utility

The java file Utils.java should be modified to change the agency name, username, and password. These should be updated with the actual agency, username and password sent from the RIITS system Administrator in the “Welcome to RIITS” email.

4.5 Build & Run Sample Code

At this point, Build the system by executing the make file. The make must be successful before continuing. Once the system is successfully built, run the sample code.

```
make          <builds the sample code>
```

```
make run     <runs the sample code>
```

Sample code has been successfully built, and is properly running, if the xml file(s) are being updated in the “xml” directory.



5.0 RIITS – CONNECTING WITH MICROSOFT .NET

This section describes how to use/modify the sample code to access RIITS data if using Microsoft .NET Visual Studio for Web Services.

5.1 Environment Setup

If using Microsoft .NET, download the appropriate sample code. After downloading the appropriate sample code, unzip the file. The top directory of this C# project is ConsoleApplication2. Open the C# project ISP_WS_test. View this project with “Solution Explorer” by clicking “view” from the menu bar and then select “Solution Explorer”. Figure 5-1 is a sample of the project.

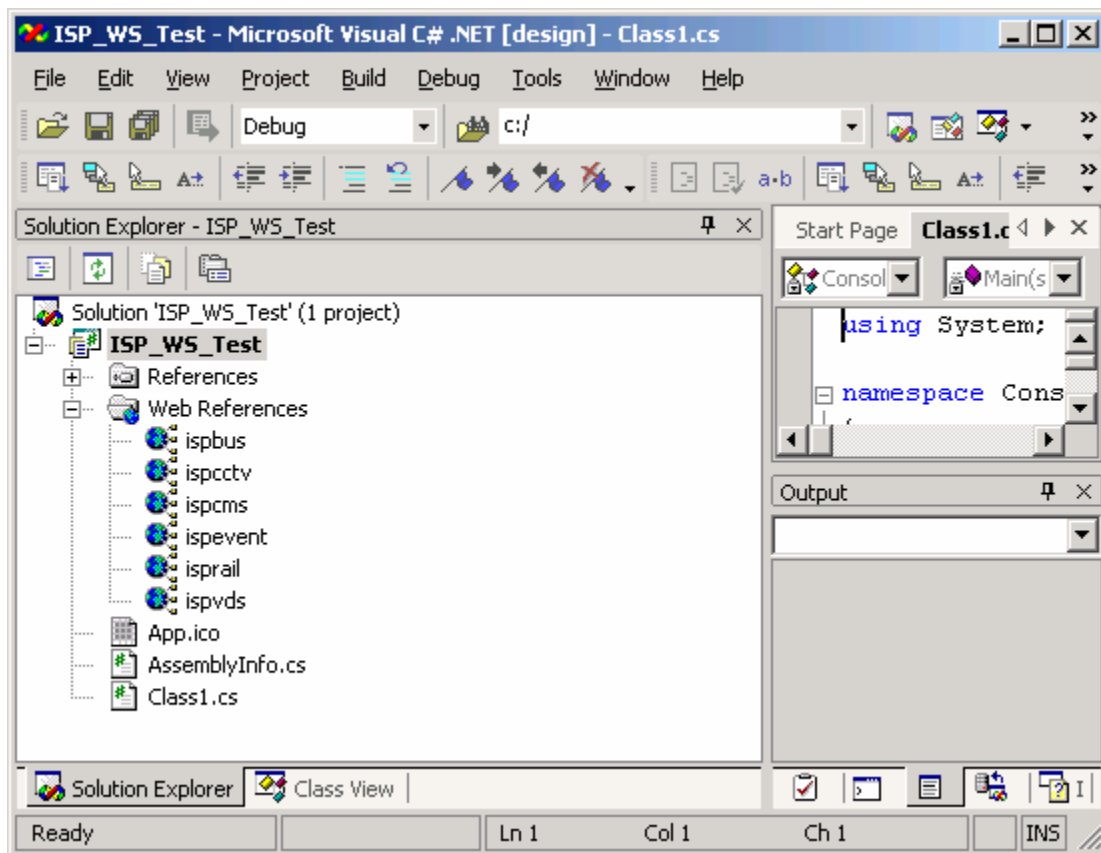


Figure 5-1: Microsoft .NET Project

5.2 Update Web Reference

Update the appropriate “Web References” in this project. Use the URL for the RIITS Web Service. To update the web reference for ispvds for example, which is the code name space for ISP_CongestionService, right click on ISPvds and then select “Properties”. Modify the Web Reference URL with the correct IP address, which was dynamically retrieved in the above WSDL section. Figure 5-2 displays how to modify the Web Reference URL.



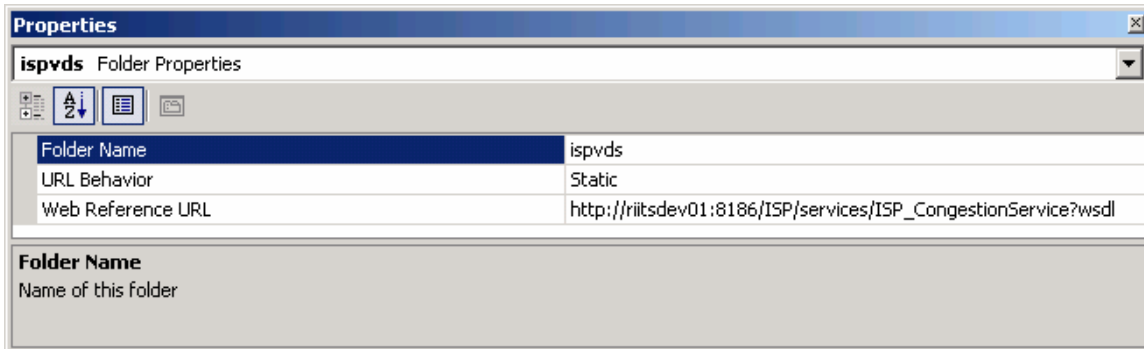
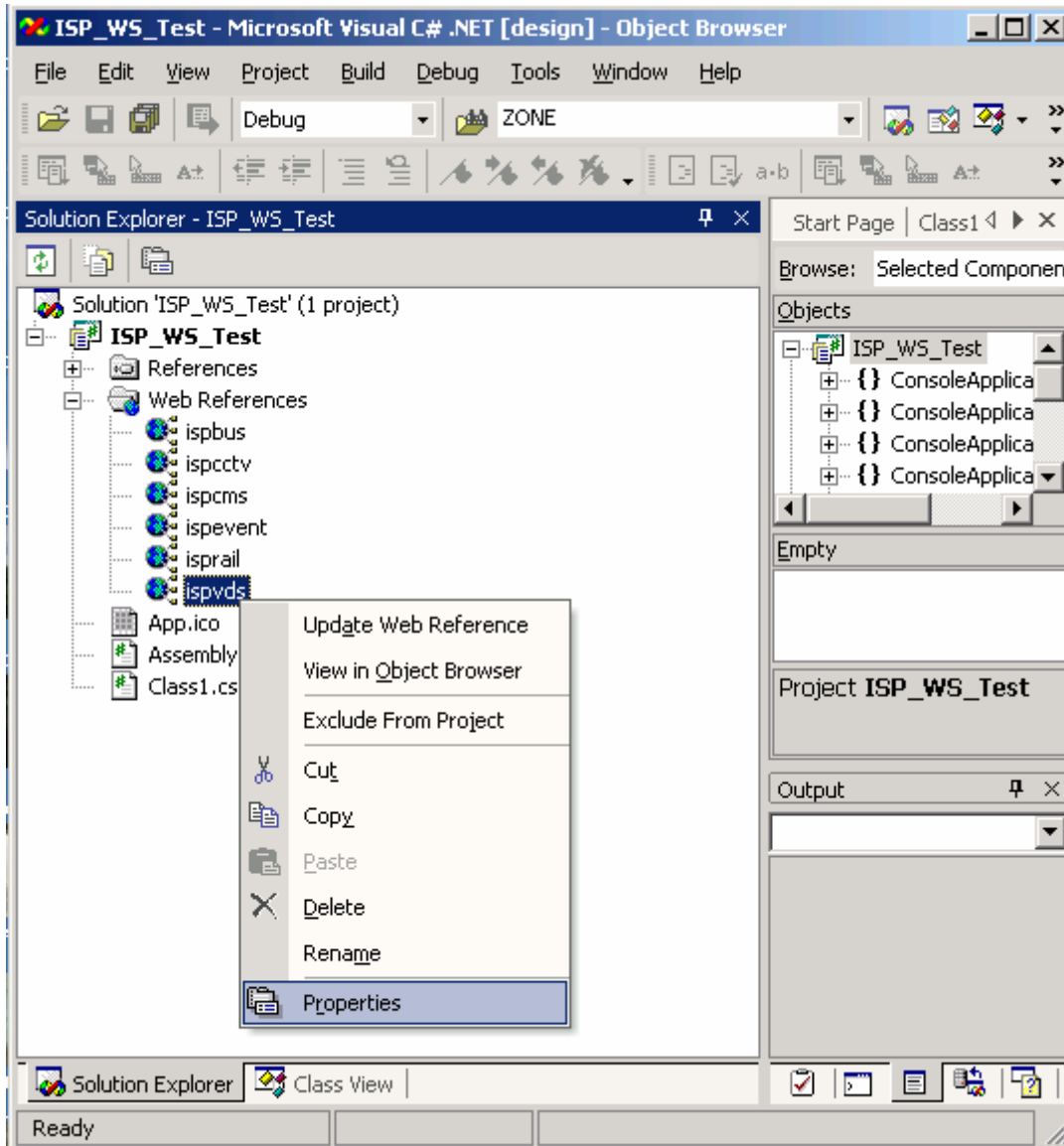


Figure 5-2: Updating Web References



5.3 Modify Class1

Modify the Class1.XmlUtil to change the userAgency, userID, and userPassData to the agency name, user name and password received from the RIITS Administrator in the “Welcome to RIITS”, email.

5.4 Build & Run Sample Code

Build and Run the project by the usual mechanism to build a project in Visual Studio. The project must successfully build with no errors. Sample code has been successfully built and run if the xml file(s) are being updated in the “xml” directory



APPENDIX A - LIST OF ACRONYMS AND TERMS

TERM	DEFINITION
AASHTO	American Association Of State Highway And Transportation Officials
ATIS	Advanced Traveler Information System
ATMS	Advanced Transportation Management Center
ATSAC	Advanced Traffic Surveillance And Control
AVL	Automatic Vehicle Location System
BOC	Bus Operations Center
CCTV	Closed-Circuit Television
CHP	California Highway Patrol
CMS	Changeable Message Sign
Congestion	Congestion identifies a section of roadway for which average speed and travel times are available.
Event	An event includes both planned and unplanned occurrences that affect transportation facilities. Planned events include: maintenance/construction lane closures and special events. Unplanned events include: emergency lane closures, incidents and accidents. Advisories are also considered events.
Event Data	Event data includes as applicable: event ID, status, location, lanes blocked, start time, estimated duration, owning agency, contact name, responders on scene, response plan information, number of fatalities, number of injuries, number of vehicles involved, and a description.
FHWA	Federal Highway Administration
GPS	Global Positioning System
HAZMAT	Hazardous Materials
HOV	High Occupancy Vehicle
Incident	An incident is an unplanned event that negatively affects transportation facilities. Incident and accident are used synonymously.
IE	Internet Explorer
ISP	Information Service Provider
ITE	Institute Of Transportation Engineers
ITS	Intelligent Transportation System
LADOT	Los Angeles Department Of Transportation



TERM	DEFINITION
LRMS	Location Reference Message Specification
NEMA	National Electrical Manufacturers Association
RIITS	Regional Integration Of Intelligent Transportation Systems
SAE	Society Of Automotive Engineers
Static Transit Data	Static transit data is data that changes relatively infrequently. Static transit data includes information regarding: routes; stops; and schedules. Transit includes both bus and rail.
TCIP	Transit Communications Interface Profiles
Timepoint	A list of locations with arrival and departure times. Note that there may be non-station locations in the list.
TMC	Transportation Management Center
TMDD	Transportation Management Data Dictionary
TOC	Transportation Operations Center
TOS	Traffic Operations System
TSC	Transit Standards Consortium
TSM	Traffic Systems Management
TSM	Transit Management System
VDS	Vehicle Detector Stations are located on roadways to detect cars and to determine the volume and occupancy at that location. Speed is derived from the volume and occupancy. VDS data is passed as part of the Congestion Web Service.
VDS Data	VDS data includes station level volume, occupancy, speed, location, last updated time, and detector status.
VMS	Variable Message Sign
WSDL	Web Services Definition Language
XML	EXtensible Markup Language

