



Knowledge to Shape Your Future

Itron Enterprise Edition™

Web Services User Guide

v7.0 SP4

Identification

Web Services User Guide
8/2/2011
v7.0 SP4

Copyright

© 2009-2011 Itron, Inc. All rights reserved.

Confidentiality Notice

The information contained herein is proprietary and confidential and is being provided subject to the condition that (i) it be held in confidence except to the extent required otherwise by law and (ii) it will be used only for the purposes described herein. Any third party that is given access to this information shall be similarly bound in writing.

Trademark Notice

Itron is a registered trademark of Itron, Inc.

All other product names and logos in this documentation are used for identification purposes only and may be trademarks or registered trademarks of their respective companies.

Suggestions

If you have comments or suggestions on how we may improve this documentation, send them to TechnicalCommunicationsManager@itron.com

If you have questions or comments about the software or hardware product, contact Itron Technical Support:

Contact

- Internet: www.itron.com
- E-mail: support@itron.com
- Phone: 1 877 487 6602

Contents

Chapter 1 Before You Begin	1
IEE Version and Web Services	1
Audience.....	1
Prerequisites.....	1
Related Documents.....	1
Terminology.....	2
 Chapter 2 Service Types and Service Hosting.....	 3
Service Types.....	3
Hosting Public Services.....	4
Hosting Internal Services	4
 Chapter 3 Service Documentation	 7
Service Documentation Overview	7
Browsing a Web Service Hosted in IIS	7
Browsing a Web Service Hosted on the Application Server	7
 Chapter 4 Service Versioning.....	 9
Service Versioning Overview	9
Versioning Policy.....	9
Breaking Changes	9
Non-Breaking Changes.....	9
Versioning Scheme	9
XML Namespaces.....	10
Version Number	10
 Chapter 5 Transports	 11
 Chapter 6 Security.....	 12
Transfer Security	12
Authentication.....	12
Authorization.....	13
 Chapter 7 Service Endpoints.....	 14
InProcess Endpoint	14
Intranet Endpoint	14
DMZToIntranet Endpoint	15
Basic Endpoint	15
Internet Endpoint	15

Custom Endpoints	16
Chapter 8 Web Service Configuration	17
Web Service Settings	17
Service Management	19
Special Configuration for IIS-Hosted Public Services	20
Configuring the IIS Application Pool.....	23
To configure the IEE application pool	23
About the Identity Tab.....	24
Special Configuration for Http Endpoints on the Application Server.....	24
Chapter 9 Service Interface Paradigms	29
Synchronous vs. Asynchronous Services	29
Web Services vs. File Imports.....	29
RequestBase	29
ResponseBase	30
BatchOperations.....	31
DateTime Fields	31
XML Namespaces	32
Chapter 10 Accessing Services	33
Configuring IEE Applications to Access Internal Services	33
Configuring Access to Services Running on Application Server	33
Overriding Configuration for a Specific Service	34
Configuring Access to MEX Endpoint.....	34
Configuring Web Services Global Timeout Setting.....	35
Accessing IEE Web Services from External Clients	35
Configuring IEE to Access External Services	37
Scenario: Asynchronous results delivery	37
Scenario: Publishing log events and device events.....	37
Scenario: Contacting Itron OpenWay	37
Scenario: Mass Market Customer Care.....	38
Endpoint Configuration for Calling External Services	38
Service Registry Config Table	38
Service Endpoint Config Table	38
WCF Client Configuration	39
Auto-Download Configuration Using WS-Metadata Exchange	40
Passing Client Credentials to External Services.....	40
Chapter 11 Service Logging	42
Application Service Task Log	42
Application Service Log Files	42
Application-Specific Logging	43
WCF Service Tracing	44
Chapter 12 Standards Support.....	45
Standards Support	45

Chapter 13 Advanced Service Configuration.....	46
Custom Endpoint Configuration	46
Sample Code: Define custom service endpoint and binding	46
To add the custom endpoint to the IEE services.....	47
Using Client Certificates for Authentication.....	48
Mass Market Customer Care and Client Certificates.....	49
To configure MM CC to send client certificates	49
Additional Steps for MM CC Web Applications	51
To configure MM CC to receive client certificates	51

CHAPTER 1

Before You Begin

Itron Enterprise Edition (IEE) incorporates multiple internal and external web services to move data among various IEE and external systems. Internal web services are confined to IEE, while public web services communicate between IEE and external systems.

This document uses the term "web services" to refer to the newer IEE web services. The older services are referred to as "legacy web services" as needed.

The legacy web services are: ReadingManager, TaskAPIManager, BillManager, ConfigurationManager, CurtailmenOperations, EventService, and ValidationSetManager. The rest of the services installed with IEE are newer web services. These include but are not limited to Device Communications, Program-based Configuration, and AMI Billing Export.

This document provides a general guide to the public IEE web services. Specific documentation for each web service is found in the individual IEE application programming interface (API) guides and the annotated XSDs and WSDLs (XML Schema Documentation and Web Services Description Language) provided with each IEE installation. You will find these in the installation directory.

IEE Version and Web Services

The information in this document applies primarily to the web services that Itron released with IEE v7.0 and later. The web services released prior to IEE v7.0 were implemented using an older web services technology. Most of the information in this document does not apply to those web services.

Audience

This guide is intended for experienced IEE implementation teams.

Prerequisites

The services discussed in this document are based on Microsoft's WCF (Windows Communication Foundation) framework. There are specific fundamental terms, such as bindings and endpoints, that require some background knowledge of WCF in particular and web services in general.

Terms such TCP, HTTP, Transport Security, and SOAP are used with the assumption the reader is familiar with those as well.

A working knowledge of energy management concepts and terminology is desirable but not required.

Related Documents

IEE Configuration Guide

IEE provides associated API materials including XSDs, WSDLs, HTML versions of the schemas, and code samples. These materials are located on the IEE installation disc or installation package under \ReferenceGuides. You can also find them on your Itron customer portal.

Terminology

The IEE Web services and API guides use the following terminology.

- **API.** This guide uses the term API to refer to the complete application programming interface.
- **Element.** The IEE APIs use elements to classify data in the XML document. Elements are sometimes referred to as tags or XML tags. An IEE API element uses an opening tag and a closing tag to define where the element begins and ends. An element can contain attributes, values, and other elements.
- **Entity.** Each node in the IEE database is called an entity. For example, service points, customers, and meters are all entities. The XML APIs use elements and attributes to define the properties of each database entity.
- **Function call.** See Operation.
- **Method.** See Operation.
- **Operation.** An operation is a service that is provided by the API. The IEE APIs use operations to define the requested actions, such as adding a meter or assigning a rate. Operations are sometimes referred to as methods, services, or function calls in other programming languages.
- **Sample code.** Sample code illustrates the syntax and properties that are required to perform a typical business operation.
- **Service.** See Operation.
- **User-defined attribute (UDA).** Administrators create UDAs and assign their values in the IEE rich client.
- **Value.** The value is the definition of an attribute or element. Values can be binary (such as **true/false** or **1/0**) or one of a group of definitions (such as **Active, Inactive, or Standby**). In the example `<Meter Type="Centron" Active="True">`, **CENTRON** is the value for the Type attribute, and **True** is the value for the Active attribute.

Service Types and Service Hosting

Service Types

IEE provides public Web services and internal Web services. Public Web services are used for integration between IEE and external applications. Internal Web services are used internally by IEE applications.

Public services are exposed to the customer and installed either into IIS or hosted by the IEE Application Server. Internal services are not exposed, although they may theoretically be accessible by a user with enough technical expertise. This is not supported.

The following table summarizes the difference between public and internal services.

	Public	Internal
Testing	Tested directly using a web service testing tool	Tested indirectly by testing the application that uses the service
Documentation	Full user documentation, including WSDL, XML Schema, and an API Guide	No user documentation provided
Versioning	Versioning scheme and governance in place to support backward compatibility	No guarantee to customer of backward compatibility and customer uses the service directly at their own risk

A Web service is executable code that must be run by a Windows process. Service hosting refers to the application or process that hosts or runs the service. IEE services are hosted by multiple processes:

- **Internet Information Services.** Public services may be hosted on a Web server by Windows IIS. This is similar to how the IEE Web application code is hosted or executed.
- **IEE Application Server.** Services may be hosted on an application server by the IEE application server Windows service. (Available only for services released with IEE 7.0 or later.)
- **IEE Client.** Some of the functions in the IEE rich client (UIApplication.exe) invoke an internal service to fetch data from the database or perform other work. By default, these services are hosted inside the IEE client process. This type of service call is analogous to invoking any other function or sub-routine in the IEE client. The code that invokes the service runs inside the same process as the service itself.

Hosting Public Services

When configuring IEE, you can choose to host public services in IIS, on the IEE application server, or in both places.

IIS Hosting

- Provides management capabilities, such as application pooling and process recycling, which can improve the availability and performance of public services. Some of these capabilities are provided in a different way by the IEE application server.
- Some security configurations are easier to configure when the service is hosted in IIS.
- Enables running the web services in a different network zone than the application server.
- The web services that were originally released prior to IEE v7.0 are implemented using the ASP.NET web service technology. These web services can only be hosted in IIS.

IEE Application Server Hosting

- Allows you to manage services directly in the IEE rich client user interface.
- Services that perform data import require the ability to write out a file readable by a service running in the IEE application server. If the service is hosted in IIS, it will need permission to be able to write out to a directory that the application server can access.
- The failover of application server hosted services works the same way as the rest of the IEE v7x failover system. Network load-balancing hardware or software is required to provide true failover.
- If you are having problems with an IIS-hosted service, testing an application server hosted version of the service can help isolate whether the problem is an IIS configuration issue.

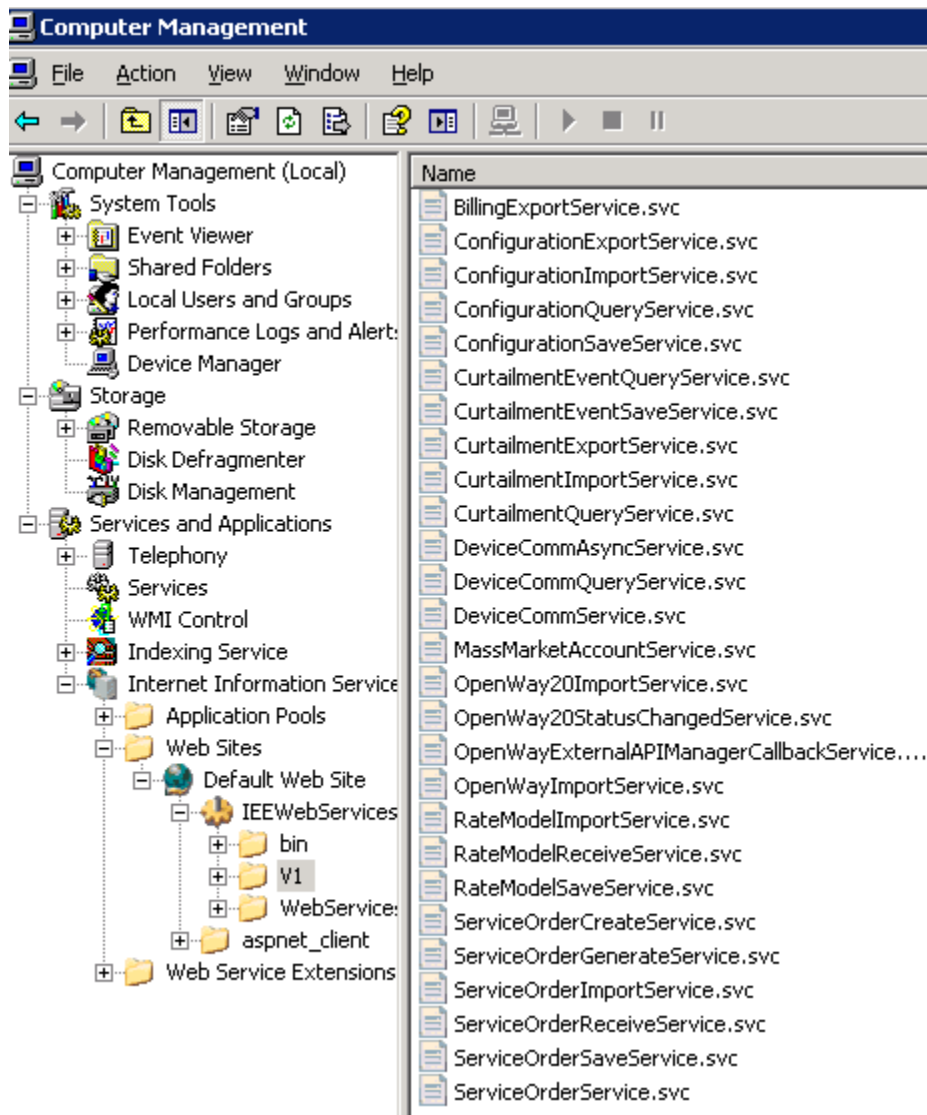
Hosting Internal Services

IEE runs the following types of internal services:

- **Utility services.** Provide fail-over and other capabilities on the application server. These run on the application server only.
- **Business services.** Import data and perform other business functions, such as fetching data for display in the Mass Market Customer Care application or sending commands to an AMI system.
- **Mass Market Customer Care services.** Must be hosted on the application server.

To host the public services in IIS

When you install IEE, select the Web Services Server feature, further described in the *IEE Installation Guide*. The services will be installed in IIS and you can view them in the Computer Management console or equivalent utility in the Windows operating system, as shown in the following illustration. (The screen may look different depending on your operating system).



To host services on the application server

1. Choose the Application Server feature when installing IEE.
2. Start the IEE Application Service windows service.
3. Open the IEE rich client.
4. Go to **System Administration > Service Management**.

5. Select one of the servers in the Server list and click **Add Service**.
6. For Service Type, select **Public Web Service Host** or **Internal Web Service Host**, as appropriate. MM CC requires that services be hosted on the application server, so for all services that will be used for MM CC, select Internal Web Service Host.
7. Enter an ID for the application service and click **OK**.
8. Click **Refresh** after a few seconds to verify that the application service has started.
9. Right-click the application service and click **View Task Log** to verify the service was able to start the individual web services. If the web services were able to start, you can see the endpoints (URLs) that are available for connecting to the service.

The following illustration shows an example of the started web services:

187759	11/17/2009 15:42 EST	Information	The ConfigurationExportService web service started successfully.
187759	11/17/2009 15:42 EST	Information	The ConfigurationExportService web service is listening on endpoint net.tcp://tal-early:8002/ConfigurationExportService/Intranet.
187759	11/17/2009 15:42 EST	Information	The ConfigurationExportService web service is listening on endpoint net.tcp://tal-early:8002/ConfigurationExportService/mex.
187759	11/17/2009 15:42 EST	Information	The ConfigurationImportService web service started successfully.
187759	11/17/2009 15:42 EST	Information	The ConfigurationImportService web service is listening on endpoint net.tcp://tal-early:8002/ConfigurationImportService/Intranet.
187759	11/17/2009 15:42 EST	Information	The ConfigurationImportService web service is listening on endpoint net.tcp://tal-early:8002/ConfigurationImportService/mex.

Service Documentation

Service Documentation Overview

Itron provides extensive documentation to help you use each IEE web service.

The IEE API guides provide general information on how to use each service. These guides are available in PDF format. They are installed with the IEE rich client application. You can also download them from Itron's customer support web site.

Annotated WSDL and XSD files are installed in the IEE bin\ServiceMetadata directory. You can also download them from Itron's customer support web site. You can download WSDL and XSD files from a running web service by navigating to the web service in a web browser.

Itron does not support WSDLs or XSDs that have been edited by customers. If you need a change or correction to a WSDL or XSD file, please contact Itron. The change can be delivered as part of an official IEE hot fix or release.

Browsing a Web Service Hosted in IIS

To browse a Web service hosted in IIS

1. In the Windows Start menu, go to **Start > Administrative Tools > Internet Information Services**.
2. In the tree, click **IEEWebServices**.
3. Open the **V1** folder. The IEE web service bootstrap files display with the .svc extension.
4. Right-click the service and select **Browse**. Your browser displays a documentation page that includes a link for downloading the WSDL. This is the WSDL for the running service. The WSDLs are also provided as part of the API documentation materials. These materials are located on the IEE installation disc or installation package under \ReferenceGuides.

Browsing a Web Service Hosted on the Application Server

To browse a Web service hosted on the application server

1. Open the IEE rich client.
2. Go to **System Administration > Service Management**.
3. Right-click the application service with **Service Type = Public Web Service Host**.
4. Select **View Task Log**.
5. Locate the startup message for the service you want to browse. After the startup is a message containing a URL. For example: the XXXXService web service is listening on endpoint http://server-name:80000/XXXXService/Basic.
6. Copy the base address portion of the URL. For example: http://server-name:8000/XXXXService.

7. Paste the base address into the browser address bar. Your browser displays a documentation page that includes a link for downloading the WSDL.

Service Versioning

Service Versioning Overview

Itron manages changes to the IEE web service interface in accordance with the versioning policy described in this chapter.

Versioning Policy

Itron may add new web services and enhance existing web services with each new IEE release or update.

Breaking Changes

Whenever possible, Itron avoids making breaking changes to existing web service interfaces. A breaking change is a change that causes the service to be incompatible with existing web service client calls.

Examples of breaking changes include:

- Adding a new required element to a request or response message.
- Removing or changing the data type of an element in a request or response message.

Itron recognizes that in some cases, it may not be possible to make necessary enhancements to a web service without introducing a breaking change to the service interface.

When a breaking change is necessary, Itron will increment the major version element to indicate the breaking change. Itron will still support the previous version. However, Itron may limit future enhancements to the web service to the new major versions.

Non-Breaking Changes

Itron strives to introduce only non-breaking changes to its web services. When a non-breaking change is necessary, Itron will increment the minor version element to indicate the non-breaking change.

Examples of non-breaking changes include:

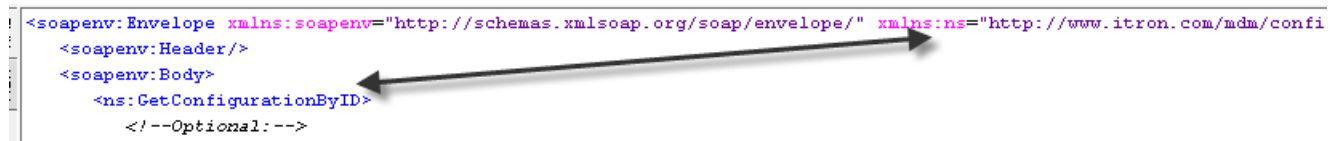
- Adding a new, optional element to a request or response message.
- Adding a new operation to a web service.

Versioning Scheme

You can verify the version of the web service by its XML namespace and the Version Number element.

XML Namespaces

In the call to a web service, the element that identifies the service operation (the immediate child of the SOAP body element) must reference the IEE namespace where this operation is defined. See the example below.



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns="http://www.itron.com/mdm/config" >
  <soapenv:Header/>
  <soapenv:Body>
    <ns:GetConfigurationByID>
      <!--Optional:-->
    
```

XML namespace references are optional in the XML that is imported using an import task, and in most of the XML that is sent to an IEE web service.

Version Number

The version number is visible in a <documentation> element at the top of the WSDL and XSD. For example:

annotation	IEE Version: 7.0.370 Service Version: V1.01
import	loc:www.itron.com/mdm.common.2008.0 ns:http://w
import	loc:schemas.microsoft.com.2003.10.Seri ns:http://sc

The version number has the following format:

Vz.nn

The **z** is the major version number and the **nn** is the minor version number. The first version of a web service is always V1.00. The major version number is part of the URL for addressing the web service. This allows multiple major versions of the web service to run concurrently.

The <documentation> element also contains the version number of the IEE release that the WSDL and XSD were delivered with. This number changes with each IEE release, even if the web service interface has not changed.

The Release Notes documentation in each IEE release describes all of the web service interface changes that are included in that release.

Transports

IEE uses multiple transport protocols to connect clients to IEE services. For example:

- **http.** An industry-standard protocol used for connectivity between IEE and other systems.
- **net.tcp.** A proprietary protocol used for connectivity between IEE clients and internal services.
- **net.pipe.** A protocol used for in-process service calls within the IEE client.

The net.tcp and net.pipe protocols have some performance benefits but require that both the client and the service are implemented using Microsoft .NET.

Security

The following security methods are required when passing messages from a client to a web service:

- **Transfer Security.** Protects the message from being viewed or changed by a third party during the transfer of the message from the client to the service.
- **Authentication.** Determines whether someone or something is who or what it is declared to be. Authentication provides proof of the identity of the client and service to each other.
- **Authorization.** Verifies that the client is allowed to access the service's functionality, or is allowed to access resources (such as data) that is requested from the service.

Transfer Security

Transfer security involves encrypting messages and applying a digital signature to them. Encryption ensures that a third party cannot view the message content. Digital signatures are used to verify that the message content was not changed during transmission.

IEE web services support two different modes of transfer security.

- **Transport-level security** using SSL and HTTPS, in which the entire connection between the client and service is encrypted. Transport-level security performs well and is interoperable with most systems. It is appropriate in deployments where there is a direct point-to-point connection between the client and the service.
- **Message-level security**, in which the contents of individual messages are encrypted and signed prior to transmission. Message-level security requires support for newer web services standards such as WS-Security and WS-Trust. These messages can be passed through intermediaries and still remain secure.

Authentication

IEE web services use server certificates to authenticate the server to the client. The certificate must be installed and configured on the server.

To authenticate the client to the server, the IEE web service processes client credentials that are passed from the client to the service. IEE web services support the following forms of client credentials:

- **Client certificates.** Preferred for “large-grained” authentication where the identity of the client is the application sending the message.
- **Username/password authentication.** Preferred for “fine-grained” authentication where a user name and password identifying the individual user that initiated a request are passed to the service.

In both cases, a user must be added to the IEE database to represent the client. If the client is being authenticated using a certificate, the Subject Name of the certificate must be pasted into the Alias field of the user configuration screen. Configure the user in the IEE rich client.

Authorization

IEE web services rely on the same role-based security model that IEE uses to control access to functionality and data. The IEE user must be associated with one or more roles and, optionally, one or more security groups. The roles define whether the client is allowed to invoke the service operation, and the security groups define what data is accessed by the service operation.

Configure roles and security groups in the IEE rich client. Refer to the rich client online help for instructions.

Service Endpoints

An endpoint is a point of contact between a client and a service. Services listen for incoming messages on one or more endpoints, and a client sends messages to one of the endpoints that a service is listening on.

The basic attributes of an endpoint include:

- **Endpoint address.** A URL that defines the network location of the endpoint.
- **Binding.** Defines the transport mechanism, security options and other characteristics of how the client and service communicate with each other.
- **Contract.** Defines the business content of the message.

IEE web services are implemented using Windows Communication Foundation (WCF). WCF has an advanced configuration capability that provides a vast array of options for configuring service endpoints to support various transports, standards, and security options. IEE distills these options into a set of standard endpoints that represent the most common cases for how clients connect to services.

Configure the standard endpoints in the IEE rich client, in System Administration > System Settings > Web Service Settings.

InProcess Endpoint

The IEE client uses this endpoint to contact services running inside the IEE client process.

- **Transport.** net.pipe
- **Transfer security.** Not applicable. The message never leaves the process.
- **Authentication.** Username and password. Performed when the user logs into the ItronEE Client.

“InProcess” is the default value for the DefaultServiceEndpointName setting in the Itron.config file.

Intranet Endpoint

IEE applications use this endpoint to contact internal services. It is enabled by default in the Web Service Settings for internal services.

- **Transport.** net.tcp. Establishes a proprietary tcp-based connection between the client and the service.
- **Transfer security.** Provided by Windows. Requires that the client and service run on the same Windows domain or two federated domains.
- **Authentication.** Username and password.

“Intranet” is the default value for the DefaultServiceEndpointName setting in the mmcc.config file.

DMZToIntranet Endpoint

IEE applications use this endpoint to contact internal services.

- **Transport.** net.tcp. Establishes a proprietary tcp-based connection between the client and the service.
- **Transfer security.** Service certificate. Allows the client and service to run on different Windows domains.
- **Authentication.** Username and password.

Basic Endpoint

External applications can use this endpoint to contact public IEE web services. IEE applications use this endpoint to contact internal services in the case where the messages must pass through a firewall.

This endpoint uses web standards that have been established for several years and is highly interoperable. It is enabled by default in the Web Service Settings for public services.

- **Transport.** http. Contents are encoded in XML.
- **Transfer security.** SSL and a service certificate.
- **Authentication.** Username and password.

Because the Basic Endpoint uses SSL, it requires additional configuration outside of IEE. This configuration is described later in this document.

Internet Endpoint

External applications use this endpoint to contact public IEE web services. IEE applications use this endpoint to contact internal services when the messages must pass through an intermediary, such as a service bus, and there is no direct point-to-point connection between the client and the server.

This endpoint relies on advanced web standards such as WS-Security, WS-Trust and WS-SecureConversation. These standards provide advanced capabilities such as message-level security.

Transport. http. Contents are encoded in XML.

Transfer security. Message-level security and a service certificate.

Authentication. Username and password.

Custom Endpoints

If the standard endpoints provided by IEE do not meet the requirements of a particular IT environment or client technology, you can define a custom endpoint using WCF configuration.

This is an XML configuration that is added to either the `InternalService.config` file or the `PublicService.config` file.

See <http://msdn.microsoft.com/en-us/library/ms733830.aspx> for more information on performing WCF configuration.

CHAPTER 8

Web Service Configuration

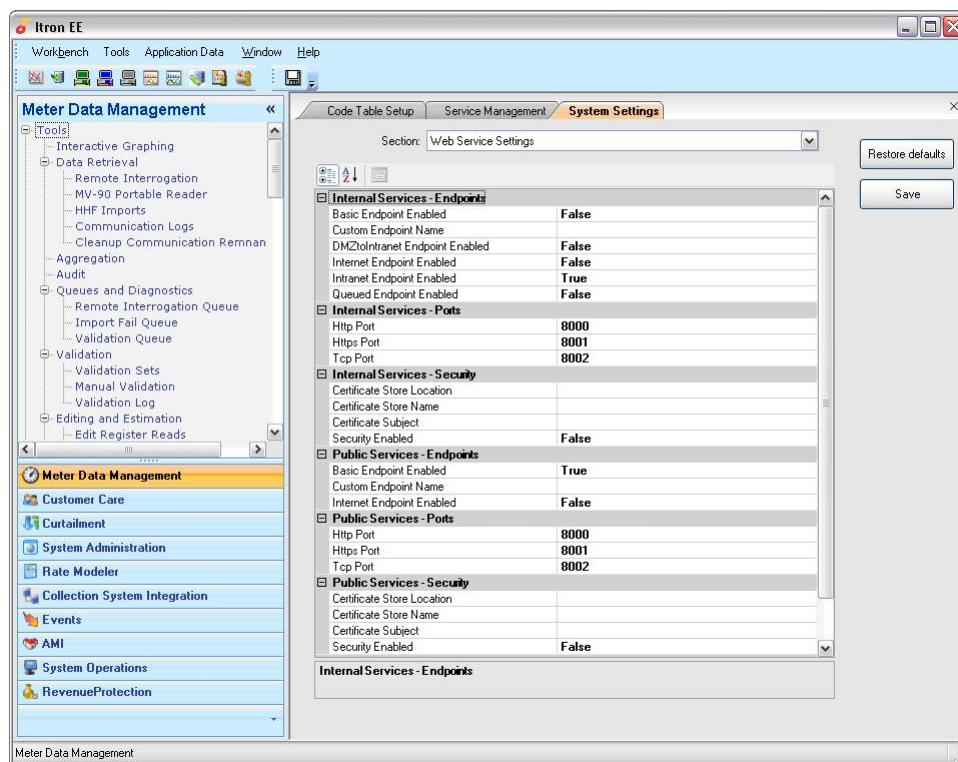
Before you can use the web services, you must configure IEE to recognize them. This occurs in the IEE rich client user interface in two sections: Web Service Settings and Service Management.

Web Service Settings

The web service settings define global properties for the IEE web services. Use Web Service Settings to enable or disable standard endpoints or to define a custom endpoint. You can also configure which ports are used in the endpoint address.

To configure web service settings

1. Go to **System Administration > System Admin > System Settings**.
2. From the Section drop-down menu, select **Web Service Settings**.



The public service ports apply only to public services that are hosted on the application server. Public services hosted in IIS use the standard ports that IIS uses for http traffic (80 and 8080).

The Security Enabled flag applies to the Basic, Internet, and DMZtoIntranet endpoints. If security is not enabled, these endpoints receive un-encrypted messages without any transfer security. This is only appropriate when a system is initially being deployed and tested.

When Security Enabled = True, a service certificate must be identified in the Certificate Store Location, Certificate Store Name, and Certificate Subject fields.

Valid values for Certificate Store Location:

- **CurrentUser.** Defines the X.509 certificate store used by the current user
- **LocalMachine.** Defines the X.509 certificate store assigned to the local machine
- If this field is left blank, the default value is **LocalMachine**.

Valid values for Certificate Store Name:

- **AddressBook.** Defines the X.509 certificate store for other users.
- **My.** Defines the X.509 certificate store for personal certificates.
- **TrustedPeople.** Defines the X.509 certificate store for directly trusted people and resources.
- **TrustedPublisher.** Defines the X.509 certificate store for directly trusted publishers.
- If this field is left blank, the default value is **My**.



Note The Security Enabled and Certificate fields do not apply to the Intranet endpoint because it uses an Active Directory token for transfer security. This endpoint is always secure without requiring a server certificate.



Note A given service can only listen on one endpoint per transport. So, for example, you cannot enable both the Basic endpoint and the Internet endpoint for public services, since they both use http as the transport.

Use the Custom Endpoint Name fields to generate a custom endpoint for each service based on WCF configuration. See [Custom Endpoint Configuration](#) on page 46.

Scrolling down in this user interface will reveal two additional settings:

- **Service Availability Check Enabled.** When the IEE client is configured to access services on the application server, enable this option if you want the client to periodically ping the application server to verify that the services are available. If a ping fails, the client notifies the user that the service is not available.
- **Number of Minutes Between Service Availability Checks.** This field controls how frequently the IEE client pings the application server.

Service Management

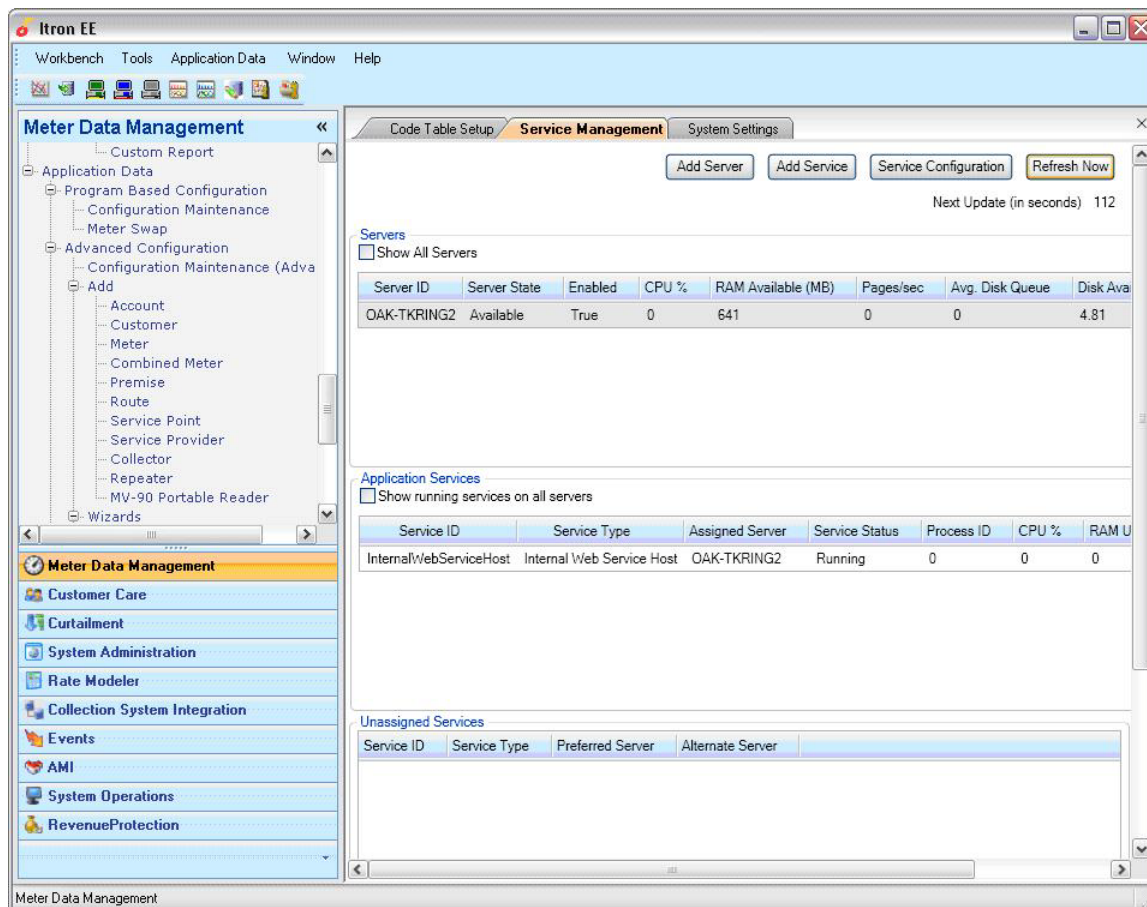
Use Service Management to define application services and assign them to servers.

An application service is a long-running process that runs on a server. Each application service runs as a separate IeeServiceRunner.exe process on the application server. The service type defines the kind of work that the application service performs.

An application service with the Internal Web Service Host or Public Web Service Host service types starts and runs IEE web services inside that application service. After a web service is started inside an application service, it starts listening for incoming messages on the endpoints that have been enabled in Web Service Settings.

To configure application services

1. Go to **System Administration > System Admin > Service Management**.



2. Select the server that will host this application service.
3. Click **Add Service**.
4. In Service Type, select **Internal Web Service Host** or **Public Web Service Host**.

Note If you choose to host the public services in IIS only, you do not need to create an application service with service type = Public Web Service Host.

5. Enter a **Service ID**.
6. Optional: Select a different **Preferred Server** to run this application service.
7. Optional: Select an **Alternate Server** to run this application service if the preferred server.
8. Click **Save**.



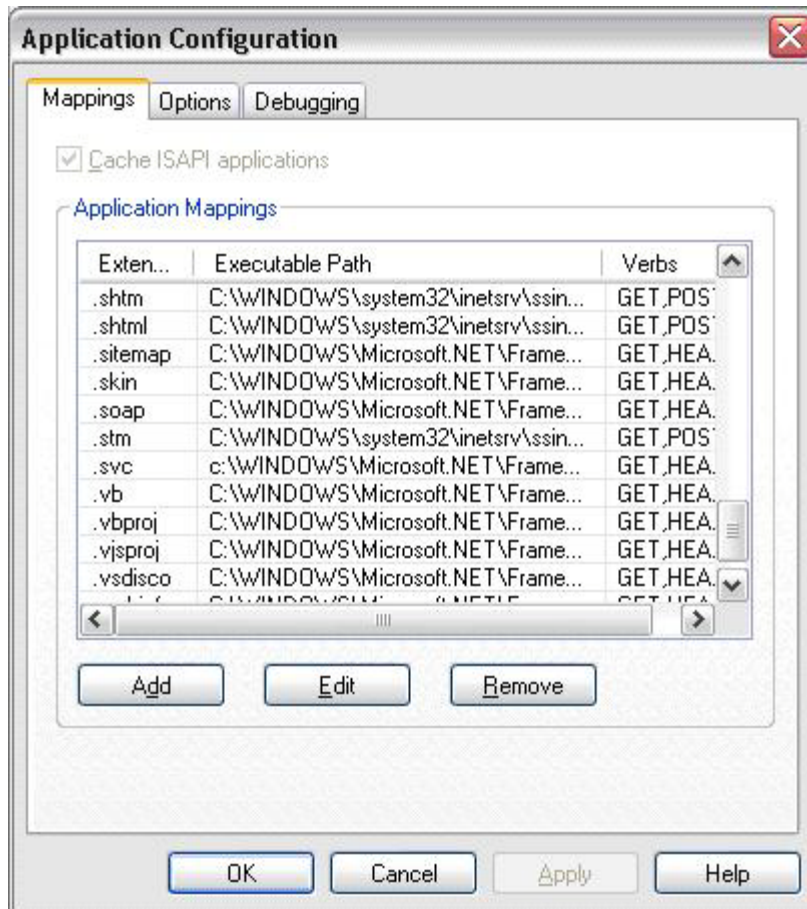
Note Once the application service is running, it ignores any changes to the Web Service Settings until the application service is restarted. To do this, click the service in the Application Services list and choose Restart from the action menu.

Special Configuration for IIS-Hosted Public Services

To enable public services hosted in IIS, you must register .svc file types and enable SSL security.

To register .svc files as a file type to be processed by IIS

1. In Windows Explorer, right-click **My Computer** and select **Manage**.
2. In the tree control in the left pane, go to **Services and Applications > Internet Information Services > Web Sites – Default Web Site > IEE Web Services**.
3. Right-click **IEE Web Services** and select **Properties**.

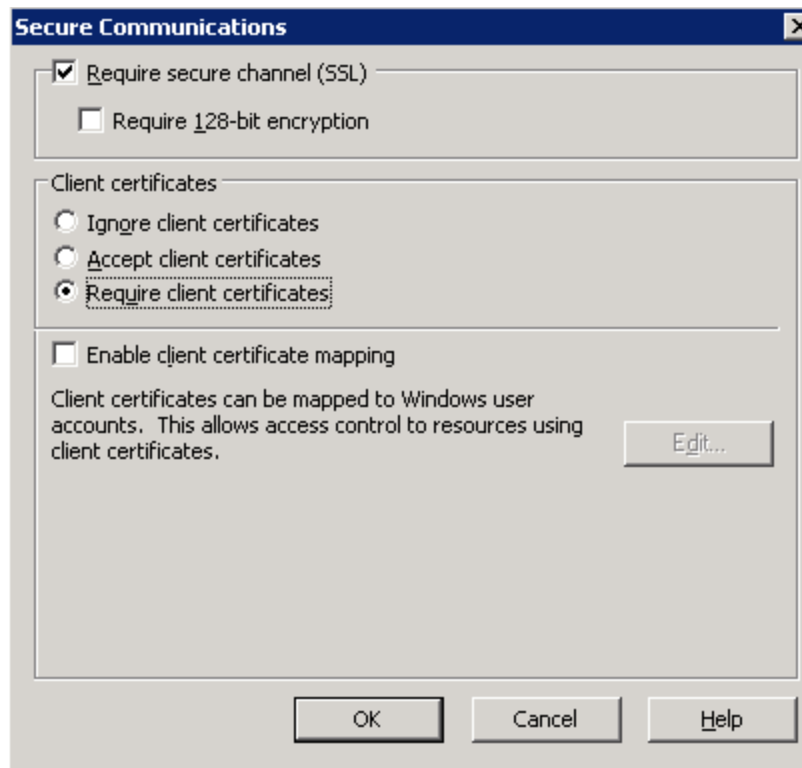
4. Click **Configuration**.

5. Scroll through the list to check whether an entry exists for extension .svc. If an entry does not exist, click **Add**.
6. In the Executable field, enter the path to the **WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_isapi.dll** file.
7. In Verbs, select **Limit To:** and enter **GET,HEAD,POST,DEBUG**.
8. Select **Script Engine**.
9. Click **OK**.

To enable SSL security for IIS-hosted services

1. In Windows Explorer, right-click **My Computer** and select **Manage**.
2. In the tree control in the left pane, go to **Services and Applications > Internet Information Services > Web Sites – Default Web Site > IEE Web Services**.
3. Right-click **IEE Web Services** and choose **Properties**.
4. Select **Directory Security**.
5. Click **Server Certificate**.

6. Follow the prompts to select a certificate to use as a server certificate.
7. If you are going to use client certificates for authentication, do the following:
 - Click **Edit**.
 - Select **Require Secure Channel (SSL)**.
 - Select **Require Client Certificates** if all of the web services will use client certificate authentication. Select **Accept Client Certificates** if only some of the web services will use client certificate authentication.



Note You must also enter the information for this same certificate in Web Service Settings.

More information on implementing certificates in IIS can be found at <http://support.microsoft.com/kb/299875> and at <http://learn.iis.net/page.aspx/144/how-to-setup-ssl-on-iis-7/>.

Configuring the IIS Application Pool

In IIS, IEE web services are accessed via the IEEWebServices virtual directory. This virtual directory is created during IEE installation.

In IIS 6.0, all virtual directories are assigned to an application pool. The application pool allows various run-time settings to be configured for the set of virtual directories assigned to the pool. By default, the IEEWebServices virtual directory is assigned to the DefaultAppPool application pool.

Itron recommends that you create a new application pool that can be configured to meet the special requirements of IEE web services.

To configure the IEE application pool

To configure the IEE application pool

1. In Windows, open the **Internet Information Services Manager**.
2. Expand **Application Pool** to see which pool the web service virtual directory is assigned to. If the virtual directory is assigned to DefaultAppPool, create a new application pool and assign it to the IEEWebServices virtual directory.
3. Right-click the new application pool and select **Properties**.
4. On the Identity tab, set the identity to a dedicated domain account. (See [About the Identity Tab](#) on page 24.)
5. Configure the performance and health characteristics of the application pool. These settings vary depending on the capabilities of your server. The following recommended settings improve the performance, reliability, and resource utilization of the IEE web services:

Tab	Field	Recommendation
Recycling	Recycle worker process	Check, and set a number of requests to 5000
Recycling	Maximum virtual memory	1000 megabytes
Performance	Web garden - maximum number of worker processes	5

Note If you are running the web services and web applications on the same server, you should assign the **Max number of worker processes** parameter to different application pools. This instruction is not intended for web applications where the **Max number of worker processes** parameter should be set to 1.

6. In Windows, open **Computer Management**.
7. Navigate to **Local Users and Groups > Groups**.
8. Edit the **IIS_WPG** group, adding the domain account that was assigned to the application pool.

About the Identity Tab

Configuring the Identity tab is important if you are using the IEE import web services or are using Windows security to connect to SQL Server.

Import services (for example, ConfigurationImport) use the file system to persist incoming data for the task system to process asynchronously. The task system accesses the import directory as a network file share. Each task template contains an Import Directory parameter where you enter the path to the appropriate directory. The application pool will therefore need to run with an identity that has access to the network file share. Grant the account that you entered on the Identity tab access to the network file share.

If you are using Windows security to connect to SQL Server, the web services will use the account configured on the Identity tab to log in to SQL Server.

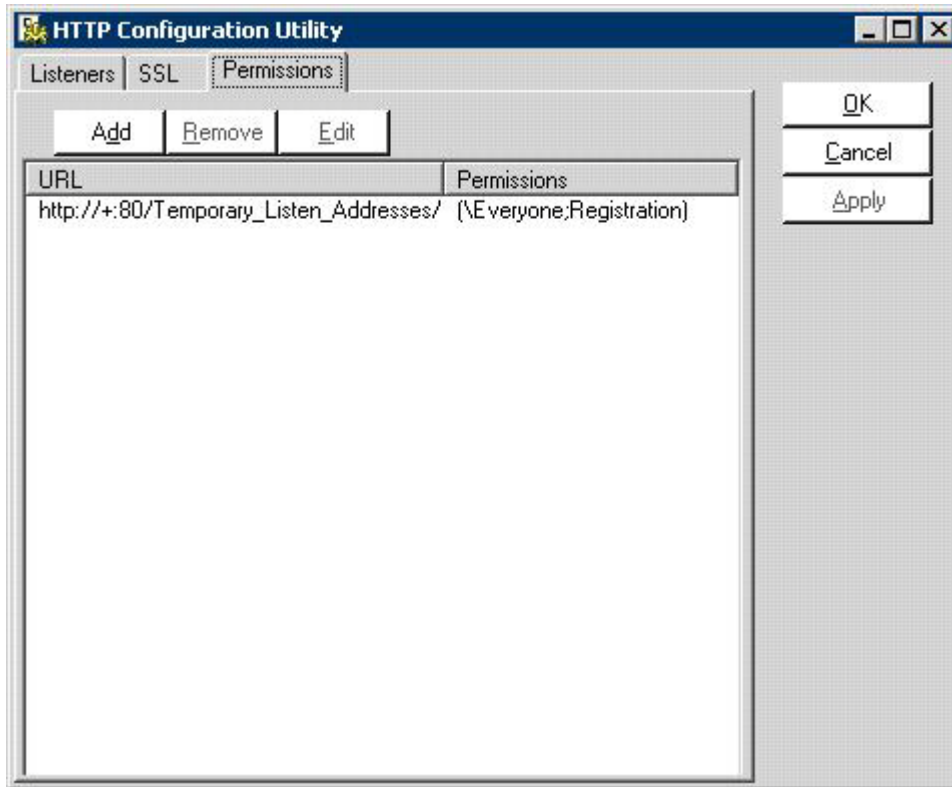
Special Configuration for Http Endpoints on the Application Server

If you enable the Basic or Internet endpoints for internal services or for public services that are hosted on the application server, you must grant access to the application to run http on the ports specified in the Web Service Settings. You must also configure SSL security.

To grant http access

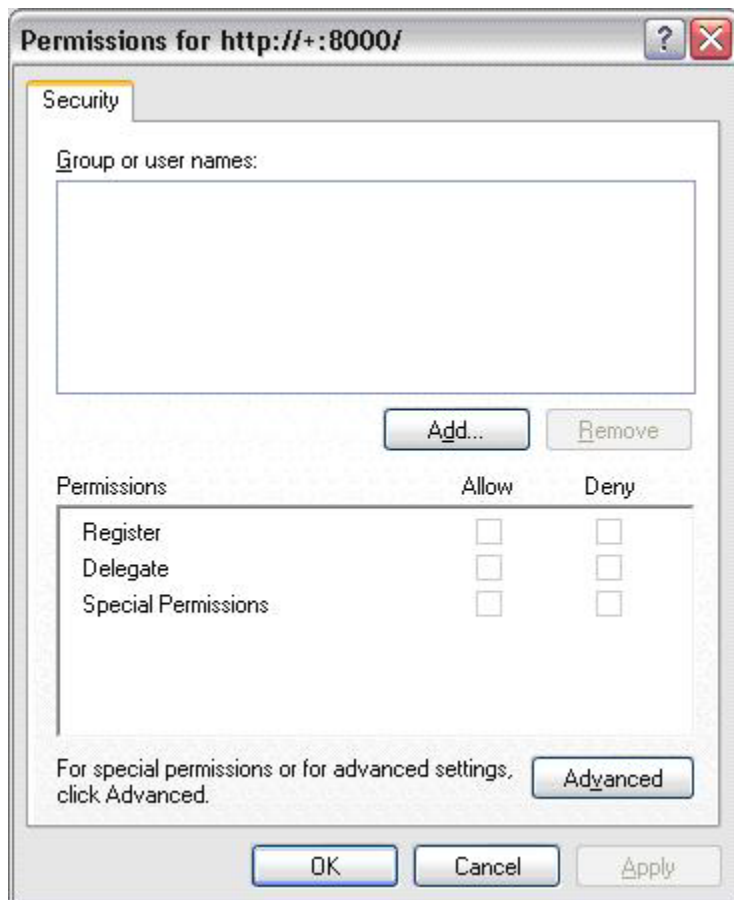
1. In Windows Explorer, go to **Program Files > Itron > Enterprise Edition > Tools**.
2. Double-click **HttpConfig.exe** to start the Http Configuration Utility.

3. Select **Permissions**.



4. Click **Add**.
5. Enter the appropriate URL:
- Unsecured access: http://+:8000/

- Secured access: : https://+:8001/

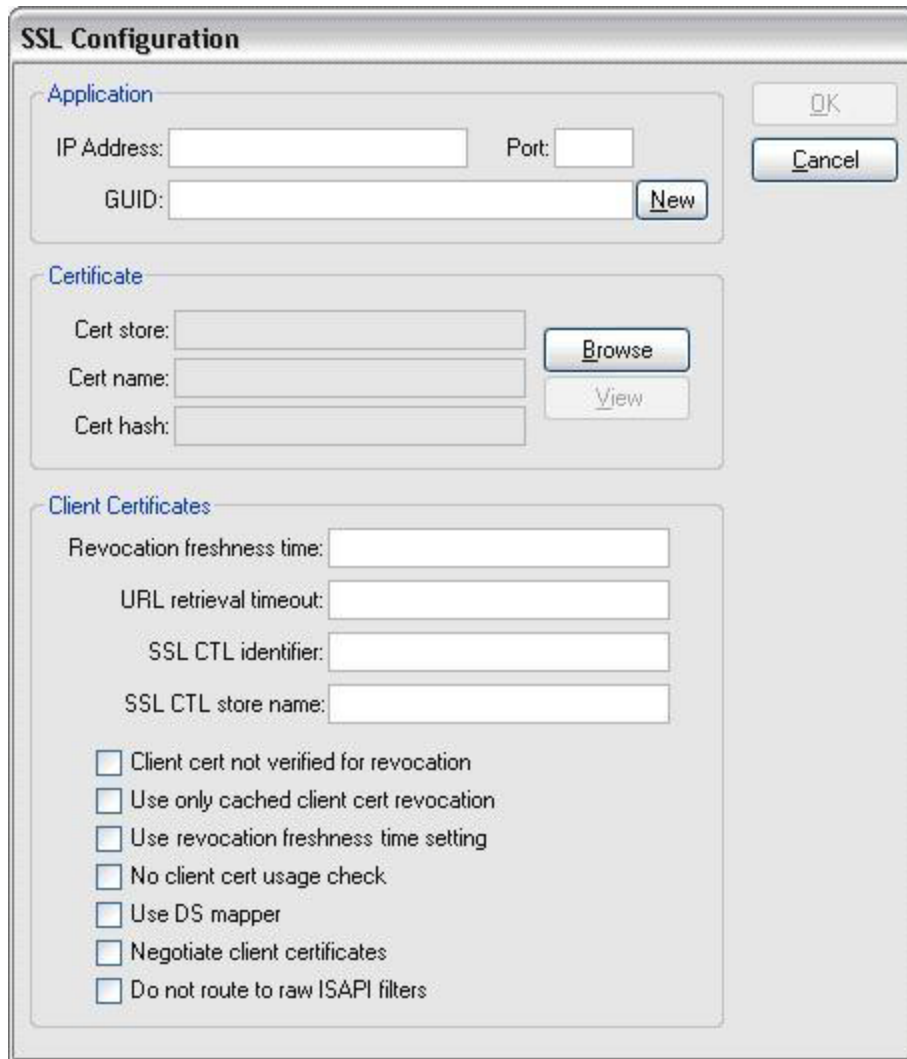


6. Click **Add**.
7. In the Users, Computers or Groups dialog box, select the windows domain account that the IEE Application Server windows service uses to log on.
8. Click **OK**.

To enable SSL for services running on the Application Server

1. In Windows Explorer, go to Program Files\Itron\Enterprise Edition\Tools.
2. Double-click **HttpConfig.exe** to start the Http Configuration Utility.
3. Select **SSL**.

4. Click **Add**.



The image shows the 'SSL Configuration' dialog box. It has three main sections: 'Application', 'Certificate', and 'Client Certificates'. The 'Application' section contains fields for 'IP Address', 'Port', and 'GUID', with a 'New' button next to the GUID field. The 'Certificate' section contains fields for 'Cert store', 'Cert name', and 'Cert hash', with 'Browse' and 'View' buttons. The 'Client Certificates' section contains four text input fields: 'Revocation freshness time', 'URL retrieval timeout', 'SSL CTL identifier', and 'SSL CTL store name'. Below these fields is a list of seven checkboxes: 'Client cert not verified for revocation', 'Use only cached client cert revocation', 'Use revocation freshness time setting', 'No client cert usage check', 'Use DS mapper', 'Negotiate client certificates', and 'Do not route to raw ISAPI filters'. On the right side of the dialog are 'OK' and 'Cancel' buttons.

5. In the IP Address field, enter: **0.0.0.0**
6. In the Port field, enter: **8001**
7. Click **Browse**.
8. Select the appropriate certificate and click **OK**.
9. If you use client certificates for authentication, select **Negotiate client certificates**.
10. Click **OK** to save the configuration.

To add the certificate to the hosts file

1. Open a text editor.
2. Open **C:\WINDOWS\system32\drivers\etc\hosts**.
3. Under the line **127.0.0.1 localhost**, enter a new line:

127.0.0.1 <certificate subject>

<certificate subject> is copied from the Subject field of the certificate that you selected with the Http Configuration utility.

Service Interface Paradigms

This section describes interface paradigms that are common across the IEE web services.

Synchronous vs. Asynchronous Services

IEE provides both synchronous and asynchronous service interfaces. These are separate services, and the service name reflects whether it is a synchronous or asynchronous service.

For example, services that save data synchronously into the IEE database are typically named XXXXXSaveService while services that asynchronously import data into IEE are named XXXXXImportService. Likewise, services that allow the client to synchronously query data from the IEE database are named XXXXXQueryService, while services that allow the client to request an asynchronous export are named XXXXXExportService.

When a client invokes a synchronous service, all processing occurs in-line with the service call, and the response indicates the final results of the processing. Asynchronous services also implement a request/response message exchange pattern, but most of the requested work is performed asynchronously after the immediate response is returned to the client. When a client invokes an asynchronous service, the request is validated, and if it is valid, IEE creates a task to process the request. The service then returns a response to indicate whether the request was accepted by IEE or to report any validation errors in the Fault and ServiceLogMessage elements.

Web Services vs. File Imports

Request messages sent to import web services can also be passed to IEE in a file by dropping the file in an IEE import directory (identified by a task template). IEE detects the file and creates a task to process the request.

IEE validates the file-based requests against the same XSD as the web service requests and applies the same business logic and processing.

RequestBase

IEE services all accept a single parameter as input. This parameter is a request element that inherits from RequestBase. RequestBase has the following sub-elements:

CorrelationID: IEE does not require, validate or perform any processing against the contents of this element. It does the following with the CorrelationID:

- It reflects the CorrelationID back in the initial response.
- It passes the CorrelationID to any downstream systems if applicable.
- It sends the CorrelationID in any asynchronous callback to a Receive service.

The client application defines the content of the **CorrelationID** and passes in a unique value if it wants to use the **CorrelationID** to match requests and responses.

ErrorProcessingInstructions: This element contains two sub-elements:

- **ReturnFaultInResponse:** If true, then the service will not use SOAP faults to return errors to the client. Instead, the presence of a Fault element in the response indicates an error. This parameter defaults to true if not specified. (In some cases, the web service infrastructure will return a SOAP fault, regardless of the **ReturnFaultInResponse** parameter. Example: when a client attempts to connect with the service in a way that violates the configured security policy.)
- **ReturnLogMessages:** If true, then the service will return additional messages that were logged during the processing of the service call in the **ServiceLogMessages** element. This parameter defaults to true if not specified.
- **ReturnLogLevel:** If **ReturnLogMessages** is true, this parameter specifies the minimum severity of the messages that will be returned. Only messages of the specified severity or higher will be returned. This parameter defaults to Warning if not specified.

Locale: This element specifies a locale string that will cause the service to return data localized to the specified locale. This generally only applies to string data, as other XML data types are locale-independent. If not specified, data is returned in the locale that is centrally configured in IEE.

ResponseBase

IEE services all return a single response element, and in most cases this element inherits from **ResponseBase**. (Some services accept and return the same element, which inherits from **TransferBase**. **TransferBase** combines the properties of **RequestBase** and **ResponseBase**). **ResponseBase** has the following sub-elements:

CorrelationID: The **CorrelationID** that was passed in the request is reflected back in this element. This is useful for correlating asynchronous responses with requests.

Fault: The presence of this element in the response indicates that the operation did not complete successfully. It has the following sub-elements:

- **CorrelationID:** The **CorrelationID** that was passed in the request is reflected back in this element. This is useful then the fault is returned as a SOAP fault.
- **Type:** The general type of fault. Possible values are:

AccessViolation: a security error occurred

InvalidParameter: the request failed initial validation

ApplicationException: a business logic error occurred

SystemException: a system problem occurred (example: database error)

Category: the fault's category. Faults are grouped into general categories such as "IDNotRecognized".

ID: the ID of the specific error that occurred

Message: a text description of the error

LogMessages: A collection of ServiceLogMessage elements. ServiceLogMessages are messages of varying severity that are logged during the execution of the service call. ServiceLogMessage has the following sub-elements:

- **Severity:** The message's severity. Possible values are CriticalError, Error, Warning, Information, Debug and Trace.
- **Category:** The message's category. Messages are grouped into general categories such as "IDNotRecognized".
- **ID:** The specific ID of the message
- **Message:** A text description of the message
- **Source:** Indicates the sub-system where the message originated

BatchOperations

Many IEE services provide operations that have a batch interface. Batch interfaces allow multiple sub-requests to be passed on one call to the service, as sub-elements of a batch-level request element. Each sub-request is transacted individually, and the service may successfully process some sub-requests while rejecting others. The results are reported by passing back a batch response containing multiple sub-responses. Each sub-response may contain a Fault element to indicate that it failed processing. Each sub-request will generally have allow a unique CorrelationID to be passed in, and these will be reflected back in the sub-responses to allow the client to correlate the sub-requests with the sub-responses.

DateTime Fields

The web services contain various fields that have an XML data type of DateTime.

XML datetimes have an optional time zone suffix that either specifies that the time is in UTC or provides an offset from UTC. The IEE web services use this information to correctly interpret DateTime fields in incoming messages.

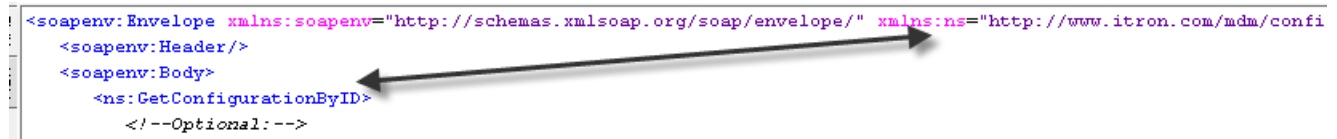
If the time zone suffix is missing, IEE assumes the time is UTC. In messages returned from IEE web services, DateTime fields always contain UTC times with the Z suffix.

Proper formatting of date/time values for this type is described in the W3C XML schema specification found at:

<http://www.w3.org/TR/xmlschema-2/>

XML Namespaces

In the call to a web service, the element that identifies the service operation (the immediate child of the SOAP body element) must reference the IEE namespace where this operation is defined. See the example below.



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns="http://www.itron.com/mdm/confi">
  <soapenv:Header/>
  <soapenv:Body>
    <ns:GetConfigurationByID>
      <!--Optional:-->
    
```

The image shows an XML snippet within a code editor. The snippet is a SOAP envelope. The first line is the opening tag for the envelope, which includes two namespace declarations: `xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"` and `xmlns:ns="http://www.itron.com/mdm/confi"`. The second line is the closing tag for the header: `<soapenv:Header/>`. The third line is the opening tag for the body: `<soapenv:Body>`. The fourth line is the opening tag for the service operation: `<ns:GetConfigurationByID>`. The fifth line is a comment: `<!--Optional:-->`. Two arrows are drawn on the image: one points from the `xmlns:ns` declaration in the first line to the `ns:` prefix in the fourth line, and another points from the `xmlns:soapenv` declaration in the first line to the `soapenv:` prefix in the third line.

XML namespace references are optional in the XML that is imported using an import task, and in most of the XML that is sent to an IEE web service.

Accessing Services

This chapter describes how to access IEE web services.

Configuring IEE Applications to Access Internal Services

The IEE client and IEE Mass Market Customer Care (MM CC) make service calls to internal services. Configure the connections from these applications to internal services in the Itron\Enterprise Edition\bin\Itron.config and Itron\Enterprise Edition\Web\MMCC\mmcc.config files respectively.

The following settings control this behavior:

- `<add key="DefaultServiceAddress" value="" />`
- `<add key="DefaultServiceEndpointName" value="" />`

To configure an application to access internal IEE web services

1. In a text editor, open **Itron > Enterprise Edition > bin > Itron.config** (for the client) or **Itron > Enterprise Edition > Web > MMCC > mmcc.config** (for MM CC).
2. Change the value of the **DefaultServiceAddress** setting to the network name or IP address of the application server that is running the Internal Web Service Host service.
3. Change the value of the **DefaultServiceEndpointName** setting to the name of one of the endpoints that have been enabled for internal services in Web Service Settings.
4. Save the changes and exit the file.

Configuring Access to Services Running on Application Server

By default, the services used by the IEE client run inside the IEE client process and so the default configuration values do not need to be changed.

```
<add key="DefaultServiceAddress" value="localhost" />
```

```
<add key="DefaultServiceEndpointName" value="InProcess" />
```

However, in certain scenarios, it may be necessary to configure the IEE client to access the services running on the application server instead of the in-process services.

To configure access to services running on the application server

1. In a text editor, open **Itron > Enterprise Edition > bin > Itron.config**.
2. Change the value of **DefaultServiceAddress** to the application server.
3. Change the value of **DefaultServiceEndpointName** to point to the required endpoint.
4. Save the changes and exit the file.

After installing IEE, configure the connection from MM CC to the services.

1. In a text editor, open **Itron > Enterprise Edition > Web > MMCC > mmcc.config**.
2. Change the value of **DefaultServiceAddress** to the application server.
3. Leave the value of **DefaultServiceEndpointName** at **Intranet**.
4. Save the changes and exit the file.

Overriding Configuration for a Specific Service

The general Itron.config settings apply to all services that are accessed by the application. To override the configuration for a specific service, append the service contract name to the setting key, separated by an underscore.

This example configures the application to access the DeviceComm service running on server MyApplicationServer:

- `<add key="DefaultServiceAddress_IDeviceComm" value="MyApplicationServer" />`
- `<add key="DefaultServiceEndpointName_IDeviceComm" value="Intranet" />`

Configuring Access to MEX Endpoint

The client application uses DefaultServiceAddress and DefaultServiceEndpointName settings to contact a special endpoint on the service called the **MEX endpoint**. This endpoint returns information to IEE. IEE then uses this information to contact the other endpoints.

In certain IT environments, IEE may not be able to reach the MEX endpoint. In that case, use the advanced configuration capabilities of WCF to configure the client to access the services.

To configure the IEE client this way, edit the Itron.config file in the Itron\Enterprise Edition\bin directory.

To configure MM CC this way, edit the Web.config file in the Itron\Enterprise Edition\Web\MMCC directory.

More information on setting up client-side WCF configuration is available here:

<http://msdn.microsoft.com/en-us/library/ms731745.aspx>.

Configuring Web Services Global Timeout Setting

IEE clients have a timeout value that defines how long to wait for a response from the web services. If the timeout is exceeded, IEE logs an error in the task monitor. This is true even for the "in-process" services that are hosted inside the IEE rich client. The default value for this timeout is 60 seconds.

You can override this timeout globally for all services by editing the `DefaultServiceTimeout` setting in the `itron.config` file. You can override the timeout for an individual service by appending the service contract name to the setting key, separated by an underscore.

To configure the global timeout setting

1. In a text editor, open **Itron > Enterprise Edition > bin > Itron.config**.
2. Change the value of the `DefaultServiceTimeout` setting to the number of seconds IEE will wait for a response from the web services. This value must be a whole number.

For example: `<add key="DefaultServiceTimeout" value="300" />`

3. Save the changes.

To configure the timeout setting for an individual service

1. In a text editor, open **Itron > Enterprise Edition > bin > Itron.config**.
2. Locate `<add key="DefaultServiceTimeout" value="{value}" />`.
3. Add an underscore and the service contract name.

For example: `<add key="DefaultServiceTimeout_IDeviceComm" value="300" />`

4. Save the changes.

Accessing IEE Web Services from External Clients

To use an external application or tool to access a public IEE web service, you must determine the address of the web service and access the service metadata.

The external client invokes the service through one of its endpoint addresses. An endpoint address is a URI consisting of two parts, the base address and the endpoint name. For example:

`http://MyAppServer:8000/V1/DeviceCommService/Basic`

In this example, the base address is `http://MyAppServer:8000/V1/DeviceCommService` and the endpoint name is `Basic`. The base address itself is composed of the scheme (`http`, `https`), the server address (`MyAppServer`), the port number (`8000`), the service version (`V1`), and the service name (`DeviceCommService`). For services that are hosted in IIS, the base address would typically look like this: `http://MyWebServer/IEEWebServices/V1/DeviceCommService`

The port number is omitted and the name of the IEEWebServices virtual directory is appended to the server address.

The endpoint addresses of the services running on the application server can be viewed in Service Management. Right-click on the application service that is hosting the public web services (Service Type = Public Web Service Host), and choose Show Task Logs. The task log lists start-up messages for the application service and for each of the web services. Beneath the start-up message for each service is a message like this:

The DeviceCommService web service is listening on endpoint
`http://MyAppServer:8000/V1/DeviceCommService/Basic`.

Stretch the window or scroll to the right to view the whole message. You can copy the endpoint address out of this message.

If you copy the base address portion of the endpoint address and paste it into the address field of a browser, the service displays a documentation page with a link to download the service WSDL.

Public IEE web services that are hosted in IIS do not appear in the IEE Service Management user interface. To view the documentation page for these services:

1. Right-click My Computer and choose **Manage**.
2. In the tree control, navigate to **Services and Applications > Internet Information Services > Web Sites – Default Web Site > IEE Web Services > V1**.
3. In the details pane, right-click a service bootstrap file and choose **Browse** to view the documentation page.

The endpoint address can be seen in the <soap:address> element of the WSDL code that is downloaded from the service.

There are various options for downloading the service metadata. Service metadata is the WSDL (Web Service Description Language) and XSD (Xml Schema Description) documents that describe the web service interface. One option is open a browser and enter the service's base address with "?wsdl" appended into the address bar. This address can also be entered into most web service integration and testing tools.

Another option is to use the svcutil.exe tool to download the metadata from the live service. In a DOS command window, navigate to the Program Files\Enterprise Edition\Tools directory and enter the following command:

```
svcutil.exe /t:metadata /d:<output directory> <service mex address>
```

Where <service mex address> is the service base address with "/mex" appended. For example:

```
svcutil.exe /t:metadata /d:c:\temp http://MyAppServer:8000/V1/DeviceCommService/Basic/mex
```

This creates files in the current directory containing WSDL and XSD describing the service.

Files containing the metadata are also deployed on the IEE application server in directory Program Files\Itron\Enterprise Edition\bin\ServiceMetadata\.

Configuring IEE to Access External Services

IEE calls out to an external service in several scenarios.

IEE needs the address, the contract, the binding, and the client identity.

This section describes the scenarios along with information about how the address and contract are made known to IEE.

Scenario: Asynchronous results delivery

Some IEE services follow an asynchronous paradigm.

When the external client invokes the IEE web service, IEE validates and saves the request returns an initial confirmation response to the external client.

This request must include the address of the external service. IEE defines and owns the service contract interface of the external service in the Receive services.

IEE processes the request asynchronously and delivers the results to an external service.

The customer or system integrator must implement a service that conforms to this interface. The WSDL for this interface is available in the Itron\Enterprise Edition\bin\ServiceMetadata directory.

Scenario: Publishing log events and device events

IEE can push log events and device events to an external service. See the individual API guides for more information on those interfaces. The IEE Configuration Guide also contains information relevant to using log and device event publishing.

IEE defines and owns the service contract interface of the external service in its Receive services. The customer or system integrator must implement a service that conforms to this interface. The wsdl for this interface is available in the Itron\Enterprise Edition\bin\ServiceMetadata directory.

Scenario: Contacting Itron OpenWay

The IEE two-way controls system calls out to OpenWay web services to request collection system operations such as Interrogation, Interactive Read, and Ping. You can use the <DeviceCommConfiguration> section of the Itron.config file to define a default address for contacting these web services. Itron OpenWay defines and owns the service contract interface of the OpenWay service, and it is known to the IEE application. If other collection systems implement this interface, then communicating with those systems will follow the same procedure.

Scenario: Mass Market Customer Care

The IEE Mass Market Customer Care application can make a web service call out to a customer information system through the MassMarketAccount service interface.

IEE defines and owns the service contract interface of the MassMarketAccount service. The customer or system integrator must implement a service that conforms to this interface. The WSDL for this interface is available in the Itron\Enterprise Edition\bin\ServiceMetadata directory.

Endpoint Configuration for Calling External Services

IEE requires endpoint configuration and client credentials to contact an external service. Endpoint configuration defines the endpoint address (URL), transport mechanism, security policy and web service standards used to communicate with the service.

To setup the endpoint configuration for contacting an external service, do the following:

- Add a row to the Service Registry Config table for the service.
- Optionally, add a row to the Service Endpoint Config table for the service.
- Optionally, add WCF client configuration to the Itron.config file.

To maintain the ServiceRegistry and ServiceEndpoint tables, open the rich client and go to **System Administration > Code Table Setup**.

Service Registry Config Table

Enter a unique, meaningful value in the Service Registry ID field. Enter the name of the service contract interface in the Service Contract field. This is the value in the name attribute of the wsdl:portType element in the WSDL that defines the service.

```
<wsdl:portType name="ICurtailmentReceive">
```

You can define a default set of parameters for all of the services that belong to an ExternalSystem by entering "Default" in the Service Contract field.

You can either leave the Base Address field blank, or use it to define a base URL that will be pre-pended to a relative URL defined in the Service Endpoint Config table.

Select an External System ID from the list for the External System ID field.

Service Endpoint Config Table

If you are using WCF client configuration, this table is not required.

To configure the Service Endpoint Config table

1. Enter a unique, meaningful ID value in the *Service Endpoint ID* field.

2. Select a Service Registry ID from the list for the *Service Registry ID* field.
3. Enter the URL to be used to contact the service in the *Endpoint Address* field. If you entered a base URL in the Base Address field of the Service Registry Config table, then you can enter a relative address here.

For example, if you set up the following:

Base Address: `http://some-server:9000/`

Endpoint Address: `some-service-endpoint`

then IEE will attempt to contact the service at `http://some-server:9000/some-service-endpoint`

4. The *Endpoint Configuration* field is used to identify the security policy and web service standards that will be used to communicate with the service. IEE provides two standard endpoint configurations:
 - **Basic.** This is the most common and interoperable set of policies and standards. It uses http, SOAP 1.1, transport level security (SSL) and username/password authentication
 - **Internet.** This is a more advanced, less interoperable set of policies and standards. It uses SOAP 1.2, message-level security and username/password authentication.

If these standard endpoint configurations do not match the security policy and standards implemented by the external web service, then you will need to add WCF client configuration to the `Irtion.config` file and leave this field blank.

5. Optionally, enter a meaningful description of the endpoint in the *Description* field.
6. Select HTTP for the *Transport Type* field.
7. Leave the default value of Unassigned in the *Server Field*.

WCF Client Configuration

Communicating with an external web service often involves security policies and/or web service standards that are not defined by the standard Basic and Internet endpoint configurations. In that case, set up WCF client configuration in the `Irtion.config` file. This is done by adding an `<endpoint>` element within the `<system.serviceModel><client>` element in the file.

For example:

```
<endpoint address="ami/2008/10/subscriptions" binding="basicHttpBinding"
contract="ExceptionSubscriberService" />
```

In this example, the address attribute defines a relative address, which will be appended to a base address configured elsewhere. If an `EndpointAddress` is defined in the Service Endpoint Config table, this attribute can be set to an empty string.

WCF provides a very rich set of configuration options for defining communications with a web service. More information on setting up client-side WCF configuration is available at <http://msdn.microsoft.com/en-us/library/ms731745.aspx>.

Auto-Download Configuration Using WS-Metadata Exchange

A web service protocol named WS-MetadataExchange defines a mechanism whereby a web service can provide a special endpoint that a client can use to download information about the security policies and web service standards supported by the web service. If a web service provides a WS-MetadataExchange endpoint, IEE can auto-download and use the endpoint configuration, eliminating the need to set up this configuration in IEE. The only thing you have to configure is the Base Address of the service, and then IEE appends the "/mex" to this address to try to contact the MetadataExchange endpoint.

For example:

Base Address: `http://some-server:9000/`

Mex endpoint: `http://some-server:9000/mex`

Passing Client Credentials to External Services

IEE supports passing two different kinds of client credentials to external services: username/password and client certificate. The binding used to contact the service determines the kind of credential passed.

If the service's binding indicates that the client credentials will be in the form of a client certificate, the certificate to pass must be installed on the computer running the client. Then the WCF client configuration must be modified to specify the client certificate. See [WCF Client Configuration](#) on page 39.

If the service's binding indicates that the client credentials will be in the form of a username and password, IEE supports various options for determining the username and password.

One option is to associate a username and password with the system that is receiving the service call.

1. Open the rich client and go to **System Administration > Code Table Setup**.
2. In the External System table, add a row representing the application that will be receiving the service call.
3. Save the changes.
4. In the External System Attribute Name Values table, select the external system and enter values for **WebServiceUsername** and **WebServicePassword**.
5. Save the changes.

The IEE database stores the password in encrypted format.

To use this configuration, IEE must determine the ID of the external system that is receiving the call. This is done by assigning the External System ID to the Service Registry Config table, as described previously.

The logic for determining the username/password and the external system differs for each scenario.

Asynchronous results delivery

The asynchronous service interfaces allow a RequestingSystem to be passed in the original request. This field must match the ID of an External System. This is the only mechanism available for sending username/password credentials in this scenario.

Publishing log events and device events

In the Events workbench, the Event Filters user interface allows a filter to be associated with a Service Endpoint Config, which is associated with a Service Registry Config. The service registry is associated with an External System. This is the only mechanism available for sending username/password credentials in this scenario.

Contacting OpenWay

The OpenWay external system is associated with the meter or recording device in meter data management configuration.

- If the WebServiceUsername and WebServicePassword are not defined in the External System Attribute Name Values table, then IEE sends the credentials that came from the originator of the request to OpenWay.
- If the request originated from the IEE client, then this will be the username/password of the user that logged into the client.
- If the request originated from a call to the DeviceComm service by an external system, then this will be the username/password that were sent from the external system.
- If the request originated from MM CC, the same logic that is described below will be used to determine the username/password.

Mass Market Customer Care

The External System is identified in the CIS External System ID field of the Customer Care Web Settings. If no value is entered there, then the following logic and configuration determines the username/password.

First, if MM CC is being run from the IEE Customer Care web application, the username/password of the user that logged in to the web application are used. If MM CC is running as a WSRP portlet, then the ForwardSecurityCredentials setting in the Itron\Enterprise Edition\Web\MMCC\MMCC.config is used to control whether the credentials are forwarded from the WSRP service call that is sent to the portlet.

Service Logging

Use the service logs to identify and diagnose failures in IEE services.

Application Service Task Log

The application service task log helps you discover the endpoints that are available for each web service hosted on an application server. This kind of log is not available for IIS-hosted web services. After the startup message for each service, IEE logs a message for each service endpoint that displays the endpoint address (URL) to use to contact the endpoint.

To view the application service task log

1. In the IEE rich client, go to **System Administration > Service Management**.
2. Right-click any service and select **Show Task Logs**.

Services with Service Type = "Internal Web Service Host" or "Public Web Service Host" display messages related to the start-up and shutdown of the application service and each of the web services the application service hosts. You will also see any errors that occurred during the startup of a web service.

Application Service Log Files

By default, the application service log files only contain error and warning messages. To configure the system to write out informational, debug, or trace messages to the log file, change the log severity setting in the `bin\Itron.config` file.

To change the log severity setting

1. In a text editor, open **Itron > Enterprise Edition > bin > Itron.config**.
2. Under `LoggingFilters`, set the `messageSeverity` attribute to:
 - Info
 - Debug
 - Trace
3. Save the changes.

IEE writes a series of log files for each application service instance, found here:

```
<application server>\Enterprise Edition\logs\Service-<service id>.log
```

(<service id> is the value displayed in the Service ID column in the IEE rich client under **System Administration > Service Management**.)

IEE writes messages from IIS-hosted public services to:

```
<application server>\Enterprise Edition\WebService\logs\IEEWebServices.log
```

These log files have an auto-rollover capability. Once the log file exceeds 10 MB in size, IEE appends a sequential number to the log file name and starts a new log file.

If you change the log severity settings in Itron.config to debug or trace, IEE clients and services will write out the messages they send and receive to the log files, along with information about the size of the message, what user credentials are sent to the service, and whether the service operation successfully starts and completes.

You can control this logging by editing these additional settings:

```
<add key="ServiceLoggingFilter" value="IDeviceComm,IDeviceCommAsync" />
```

Adding this setting to the <appSettings> section of the Itron.config file allows you to define which services log messages. In the value attribute, enter a comma-separated list of the service contracts that you want to log messages for. In the above example, only messages sent to or received by the IDeviceComm and IDeviceCommAsync services will be logged.

```
<add key="ServiceLoggingVerbosity" value="Brief" />
```

Adding this setting to the <appSettings> section of the Itron.config file allows you to define how much information is logged for each service call. The valid values are as follows:

- **Brief.** Information about the service call is logged, but the actual message is not logged. This is the default behavior if this setting is not provided.
- **Full.** The messages that pass between the client and service are logged in the log file.
- **None.** No service calls are logged. Use this when you have enabled debug logging for other purposes and don't want to see any logging related to service calls.

By default, the first 524288 bytes (.5 MB) of each message is written out to the log. Use this setting to override the amount of data logged to the specified number of bytes. In the example below, the first 1000 bytes of each message will be logged.

```
<add key="ServiceLoggingMaxMessageSize" value="1000" />
```

Application-Specific Logging

IEE writes application-specific logs that track specific kinds of activity. These logs typically track both API and user interface activity. You can view these in the IEE rich client under **System Operations > Task Monitor**.

WCF Service Tracing

Each IEE installation includes a WCF Service Trace Viewer to track the detailed communication between external applications and IEE web services. WCF service tracing can track both the client calls to a web service and the server activity in receiving and processing the client calls.

Most of this information is more easily viewable in the application service log files, but WCF traces are sometimes useful in diagnosing the root cause of web service failures.



Caution Do not enable WCF Service Tracing unless absolutely necessary. File size grows exponentially. If an external system is sending requests without any errors, but IEE is not receiving the requests, enabling WCF Service Tracing might be necessary for troubleshooting. However, disable WCF Service Tracing as soon as the troubleshooting is complete.

The IEE config files include WCF service tracing information, but it is commented out and disabled by default. To trace, you must enable the WCF service tracking.

To enable WCF service tracing

Remove the comment tags around the service.diagnostics element. (Search for "<!-- Uncomment to write to trace/message log" to find the section.)

To trace...

- **Server activity for application-server-hosted web services**, edit Enterprise Edition\bin\IeeServiceRunner.exe.config. IEE writes this trace information to Enterprise Edition\logs\ItronServiceHostTrace.svclog.
- **Server activity for IIS-hosted web services**, edit Enterprise Edition\Web\Web.config. IEE writes this trace information to Enterprise Edition\Web\Service\ItronServiceHostTrace.svclog.
- **Client calls from MM CC in portlet mode**, edit Enterprise Edition\Web\MMCC\Web.config. IEE writes this trace information to Enterprise Edition\logs\ItronServiceClientTrace.svclog.
- **Client calls from Commercial & Industrial Customer Care or MM CC web application**, edit Enterprise Edition\Web\Web.config. IEE writes this trace information to Enterprise Edition\logs\ItronServiceClientTrace.svclog.
- **Client calls from the IEE rich client**, edit Enterprise Edition\bin\UIApplication.exe.config. IEE writes this trace information to Enterprise Edition\logs\ItronServiceClientTrace.svclog.

To view trace files

1. Run <application server>\Enterprise Edition\Tools\SvcTraceViewer.exe.
2. Use the File menu to open the trace file you want to view.

If you open both a client and server trace file at the same time, you can see the flow of messages between the client and the server.

Standards Support

Standards Support

The following table lists the internet standards and protocols supported by IEE web services, as well as the IEE endpoints that support the protocol.

Internet Standards and Protocols	IEE Support
HTTP 1.1	Basic, Internet, Custom
WSDL	Basic, Internet, Custom
WS-Policy 1.1 / 1/5	All
WS-MetadataExchange	All
SOAP 1.1	Basic, Custom
WSS SOAP Message Security 1.0	Basic, Custom
WSS SOAP Message Security Username Token	Basic, Custom
WSS SOAP Message Security X.509 Certificate Token Profile 1.0	Basic, Custom
SOAP 1.2	Internet, Custom
WS-Addressing 2005/08	Internet, Custom
WSS SOAP Message Security 1.1	Internet, Custom
WSS SOAP Message Security 1.1 UsernameToken	Internet, Custom
WSS SOAP Message Security X.509 Certificate Token Profile 1.1	Custom
WSS SOAP Message Security Kerberos Token Profile 1.1	Custom
WS-SecureConversation	Custom
WS-Trust	Custom
WS-ReliableMessaging	Custom

Advanced Service Configuration

Custom Endpoint Configuration

If the standard endpoints provided by IEE do not meet the requirements of a particular IT environment or client technology, then you can define a custom endpoint using WCF configuration. This is XML configuration that is added to either the `InternalService.config` file or the `PublicService.config` file. You can define custom service endpoints and custom bindings. A custom binding can be applied to all of the web services. Defining a custom service endpoint allows you to specify which service will host the endpoint.

See <http://msdn.microsoft.com/en-us/library/ms733830.aspx> for more information on performing WCF configuration.

Sample Code: Define custom service endpoint and binding

This code sample defines a custom service endpoint named "CustomBasic" that references a custom binding named "CustomBasicBinding". This configuration causes the `OpenWay20ImportService` to listen on that endpoint, using the transport and security protocols defined in the `CustomBasicBinding` configuration.

The example also defines an additional custom binding named "ClientCert". No service endpoints in the config file reference this binding. If you enter the name of this binding into the Custom Endpoint Name field in the Web Service Settings screen, then all of the IEE web services will listen on an endpoint named with \ClientCert appended to the endpoint URL, using the transport and security protocols defined in the `ClientCert` binding configuration.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.serviceModel>
    <services>
      <service name="Itron.EE.Facade.Service.Solution.OpenWay.OpenWay20ImportService" >
        <endpoint
          name="CustomBasic"
          address="CustomBasic"
          binding="basicHttpBinding"
          bindingConfiguration="CustomBasicBinding"
          contract="IExternalAPIManagerCallback"
        />
      </service>
    </services>
    <bindings>
      <basicHttpBinding>
        <binding name="CustomBasicBinding"
          <security mode="Transport" />
        </binding>
      </basicHttpBinding>
      <wsHttpBinding>
        <binding name="ClientCert">
          <security mode="Message">
            <message clientCredentialType="Certificate"
              negotiateServiceCredential="false"
              establishSecurityContext="false" />
          </security>
        </binding>
      </wsHttpBinding>
    </bindings>
  </system.serviceModel>
</configuration>
```

To add the custom endpoint to the IEE services

1. In the IEE rich client, go to **System Administration > System Settings**.
2. Select the **Web Service Settings** section.
3. To enable this endpoint on the internal IEE services, edit the Custom Endpoint Name field in the Internal Services - Endpoints sub-section.
4. To enable this endpoint on the public IEE services, edit the Custom Endpoint Name field in the Public Services - Endpoints sub-section.
5. Click **Save**.

Using Client Certificates for Authentication

When security is enabled in IEE web services, the services expect a username and password to be passed in the WS-Security section of the incoming SOAP header. If these tokens are not passed to the service, the service reports an authentication failure to the client.

To enable IEE web services to use a client certificate instead of username/password for authentication, configure a custom endpoint using a binding configuration like this:

```
<wsHttpBinding>
  <binding name="ClientCert">
    <security mode="Message">
      <message clientCredentialType="Certificate"
        negotiateServiceCredential="false"
        establishSecurityContext="false" />
    </security>
  </binding>
</wsHttpBinding>
```

The service uses a client certificate to authenticate messages sent to this endpoint. This certificate must be linked to an IEE user record. Then, after the certificate is authenticated, the linked IEE user will be used for authorization.

To link an IEE user to the client certificate

1. In the IEE rich client, go to **System Administration > Security Setup > Security User**.
2. Click the user you want to link to the client certificate. (Create a new user first if necessary.) This displays the user properties pane.
3. Open the certificate properties and copy the Subject.
4. Paste the Subject into the User Alias field in the user properties pane.
5. Save the user record.

The service uses the security groups and roles assigned to this user to determine whether the client has access to the service operation and what IEE data the service can access.

The remaining configuration for client certificate authentication depends on whether the web services are hosted in IIS or on the IEE application server.

If the web services are hosted in IIS, then you must configure IIS to accept or require the client certificate. See [Special Configuration for IIS-Hosted Public Services](#) on page 20.

If the IEE web services are hosted on the application server, then you must select **Negotiate client certificates** in the HTTP Configuration Utility. See [Special Configuration for Http Endpoints on the Application Server](#) on page 24.

Mass Market Customer Care and Client Certificates

It is possible to use client certificates to control access between the MM CC web application or the MM CC portlets and the IEE application server. It is also possible to use client certificates to control access between the IEE application server and a CIS.

The following examples show the entire <system.ServiceModel> element. When editing the configuration files, you may need to add the relevant portions of the example into an existing <system.ServiceModel> element and/or its sub-elements.

In the following examples, the <storeLocation> and <storeName> elements identify where the certificate is deployed in the computer's certificate store. You can use the Microsoft Management Console to view the certificates installed on a computer. Information on how to view the certificates on a computer is available [here](http://technet.microsoft.com/en-us/library/cc785226%28WS.10%29.aspx):

<http://technet.microsoft.com/en-us/library/cc785226%28WS.10%29.aspx>.

The Personal store name in the Microsoft Management Console is referred to as "My" in the XML configuration. The Local Computer store location is referred to as "LocalMachine" in the XML configuration.

To configure MM CC to send client certificates

To configure a server to send client certificates

1. To configure a server to send client certificates, edit one of the following configuration files:
 - To configure the MMCC web application to send client certificates, edit the web.config file in Itron\Enterprise Edition\Web.
 - To configure the MMCC portlets to send client certificates, edit the web.config file in Itron\Enterprise Edition\Web\MMCC.
 - To configure the IEE application server to send client certificates, edit the Itron.config file in Itron\Enterprise Edition\bin.
2. Edit the configuration file and add the following configuration, replacing the **dns values** (shown in **bold text**) with the applicable values.

```
<system.serviceModel>
  <client>
    <endpoint name="ClientCert"
      address="MassMarketCCSettingsService/ClientCert"
      binding="wsHttpBinding"
      bindingConfiguration="ClientCert"
      behaviorConfiguration="ClientCert"
      contract="IMassMarketCCSettings">
      <identity>
        <dns value="DNS name of application server" />
      </identity>
    </endpoint>
    <endpoint name="ClientCert"
      address="IntervalQueryService/ClientCert"
      binding="wsHttpBinding"
      bindingConfiguration="ClientCert"
      behaviorConfiguration="ClientCert"
      contract="IIntervalQuery">
      <identity>
        <dns value="DNS name of application server " />
      </identity>
    </endpoint>
    <endpoint name="ClientCert"
      address="MassMarketAccountService/ClientCert"
      binding="wsHttpBinding"
      bindingConfiguration="ClientCert"
      behaviorConfiguration="ClientCert"
      contract="IMassMarketAccount">
      <identity>
        <dns value="DNS name of application server " />
      </identity>
    </endpoint>
  </client>
  <bindings>
    <wsHttpBinding>
      <binding name="ClientCert">
        <security mode="Message">
          <message clientCredentialType="Certificate"
            negotiateServiceCredential="false"
            establishSecurityContext="false" />
        </security>
      </binding>
    </wsHttpBinding>
  </bindings>
  <behaviors>
    <endpointBehaviors>
      <behavior name="ClientCert">
        <clientCredentials>
          <clientCertificate
            findValue="the client certificate Subject"
            storeLocation="LocalMachine"
            storeName="My"
            x509FindType="FindBySubjectName" />
          <serviceCertificate>
            <defaultCertificate
              findValue="the service certificate Subject"
              storeLocation="LocalMachine"
              storeName="My"
              x509FindType="FindBySubjectName" />
            <authentication
              certificateValidationMode="ChainTrust"
              revocationMode="NoCheck" />
            </serviceCertificate>
          </clientCredentials>
        </behavior>
      </endpointBehaviors>
    </behaviors>
  </system.serviceModel>
```

Additional Steps for MM CC Web Applications

If you are running MM CC as a web application, edit the web.config file in \Program Files\Itron

Edit the mmcc.config file and update the default service address and endpoint name values:

```
<add key="DefaultServiceEndpointName" value="ClientCert" />
```

```
<add key="DefaultServiceAddress" value="the DNS name of the IEE application server" />
```

Open the **mmcc.config** file and update the default service endpoint name value:

```
<add key="DefaultServiceEndpointName" value="ClientCert" />
```

To configure MM CC to receive client certificates

To configure IEE application server to receive client certificates from MM CC

1. Add a custom endpoint to receive the certificate (See [Custom Endpoint Configuration](#) on page 46.)
2. Map the certificate to an IEE user record. (See [Using Client Certificates for Authentication](#) on page 48.)