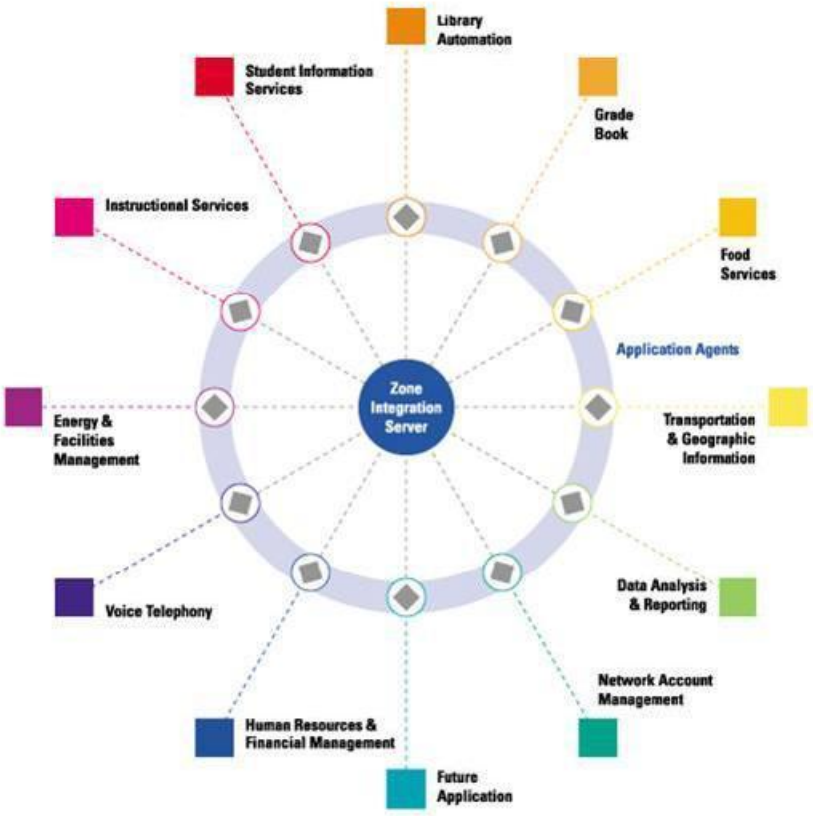
Schools Interoperability Framework™

SIF Global Web Services

Implementation Specification 2.6

July 23, 2012



Document Version: 4

This version:
http://specification.sifassociation.org/Global/2.6/ws/SIF_WebServices2p6-4.pdf

Previous version:
http://specification.sifassociation.org/Global/2.5/ws/SIF_ws2p5.pdf

1 Preamble

1.1 Abstract

1.1.1 What is SIF?

The Schools Interoperability Framework (SIF) is not a product, but a technical blueprint for enabling diverse applications to interact and share data related to entities in the pK-12 instructional and administrative environment. SIF is designed to:

- Facilitate data sharing and reporting between applications without incurring expensive customer development costs;
- Enhance product functionality efficiently; and
- Provide best-of-breed solutions to customers easily and seamlessly.

The SIF Implementation Specification defines:

- An XML-based messaging framework that allows diverse software applications to interoperate and share and report data related to entities in the pK-12 instructional and administrative environment;
- An HTTP(S)-based transport for conveying these SIF messages;
- An alternative SOAP-based transport and corresponding set of WSDL files which allow web services to fully participate in these interactions.
- An abstract, platform-independent definition of a message queue for reliable delivery of asynchronous SIF messages and related synchronous administrative functions—the *Zone Integration Server (ZIS)*; and
- An abstract, platform-independent definition of the interface between a software application and the ZIS—the *SIF Agent*.

These are known collectively as the *SIF Infrastructure*. The SIF Implementation Specification also defines the *SIF Data Model*:

- an XML-based data model that models entities in the pK-12 environment as *SIF Data Objects* to be shared between applications.

A *SIF Zone* is a distributed system that consists of a ZIS and one or more software applications with a SIF Agent (a *SIF-enabled application*) sharing/reporting one or more SIF data objects over a network. A *SIF Implementation* consists of one or more SIF Zones deployed and configured to meet customer data sharing and reporting needs.

The SIF Implementation Specification defines architecture requirements and communication protocols for software components and the interfaces between them; it makes no assumption of specific hardware or software products needed to develop SIF-enabled applications and Zone Integration Server implementations, other than their ability to support technologies leveraged as the foundation for SIF, most prominently XML and HTTP(S).

1.1.2 Schools Interoperability Framework Association

The Schools Interoperability Framework Association (SIF Association) is an industry initiative to enable interoperability and data sharing between software applications in the pK-12 instructional and administrative

environment, and the forum for companies and educators to participate in the development of SIF specifications in the SIF Association's working groups and task forces. The SIF Association is designed to:

- Join industry leaders in creating the next-generation framework for education technology; and
- Leverage co-marketing opportunities with partners and distributors.

1.2 Disclaimer

The information, software, products, and services included in the SIF Implementation Specification may include inaccuracies or typographical errors. Changes are periodically added to the information herein. The SIF Association may make improvements and/or changes in this document at any time without notification. Information contained in this document should not be relied upon for personal, medical, legal, or financial decisions. Appropriate professionals should be consulted for advice tailored to specific situations.

THE SIF ASSOCIATION, ITS PARTICIPANT(S), AND THIRD PARTY CONTENT PROVIDERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY, RELIABILITY, TIMELINESS, AND ACCURACY OF THE INFORMATION, SOFTWARE, PRODUCTS, SERVICES, AND RELATED GRAPHICS CONTAINED IN THIS DOCUMENT FOR ANY PURPOSE. ALL SUCH INFORMATION, SOFTWARE, PRODUCTS, SERVICES, AND RELATED GRAPHICS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. THE SIF ASSOCIATION AND/OR ITS PARTICIPANT(S) HEREBY DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THIS INFORMATION, SOFTWARE, PRODUCTS, SERVICES, AND RELATED GRAPHICS, INCLUDING ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT.

IN NO EVENT SHALL THE SIF ASSOCIATION, ITS PARTICIPANT(S), OR THIRD PARTY CONTENT PROVIDERS BE LIABLE FOR ANY DIRECT, INDIRECT, PUNITIVE, INCIDENTAL, SPECIAL, CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF USE, DATA, OR PROFITS, ARISING OUT OF OR IN ANY WAY CONNECTED WITH THE USE OR PERFORMANCE OF THIS DOCUMENT, WITH THE DELAY OR INABILITY TO USE THE DOCUMENT, THE PROVISION OF OR FAILURE TO PROVIDE SERVICES, OR FOR ANY INFORMATION, SOFTWARE, PRODUCTS, SERVICES AND RELATED GRAPHICS OBTAINED THROUGH THIS DOCUMENT OR OTHERWISE ARISING OUT OF THE USE OF THIS DOCUMENT, WHETHER BASED ON CONTRACT, TORT, STRICT LIABILITY, OR OTHERWISE, EVEN IF THE SIF ASSOCIATION, ITS PARTICIPANT(S), OR THIRD PARTY CONTENT PROVIDERS HAVE BEEN ADVISED OF THE POSSIBILITY OF DAMAGES. IF YOU ARE DISSATISFIED WITH ANY PORTION OF THIS DOCUMENT OR WITH ANY OF THESE TERMS OF USE, YOUR SOLE AND EXCLUSIVE REMEDY IS TO DISCONTINUE USING THIS DOCUMENT.

This specification is released with the following provisos to developers and educators.

1.3 Certification and Compliance Claims

Though a product may be demonstrated to comply with this specification, no product may be designated as *SIF Certified*™ by an organization or individual until the product has been tested against and passed established compliance criteria, published separately [SIFCertification]. Organizations and individuals that are currently paying annual membership dues to the SIF Association and dedicating resources to the SIF Association and dedicating resources to the initiative may also use the designation *SIF Participant* to describe their involvement with the SIF Association and SIF in marketing, public relations and other materials.

1.4 Permissions and Copyright

2 Introduction and Background

This volume documents how the web service technologies SOAP and WSDL are leveraged by the SIF standard to create a second reference transport for conveying SIF messages between agent/applications and the ZIS.

While this volume is dependent upon and references details contained in Volume 1 (Infrastructure) of the SIF Specification, unless otherwise noted, that volume will remain independent of the information contained here. It is anticipated that the contents of this volume will be more fully integrated into the SIF Infrastructure volume in subsequent releases.

2.1 Guiding Principles

The overarching goal of the web service mapping of the SIF Transport was to insure that the large and growing number of deployed SIF-based solutions (Zones), which were created independently of these technologies, could still incorporate them effectively in a seamless and incremental fashion without impacting day to day operations.

Adherence to the set of guiding principles below determined how the SIF architecture was extended to support web service technology. This included the addition of a new SOAP-based SIF reference transport and a set of WSDL port types to encapsulate the services provided by the ZIS.

2.1.1 Backwards compatibility must not be broken

There is a seamless and incremental web service migration path provided for all existing deployed SIF v2.x Zones.

- A "Web enabled" ZIS (v2.5 and above) is capable of supporting all v2.x agent/application pairs.
- SIF Web Applications using the new technologies are capable of being added to a SIF v2.5 and above Zone without impacting the operations of any pre-existing component.
- Existing SIF Zone components remain completely unaware of whether their "partner" (requestor, responder, publisher or subscriber) is a SIF Web Application or not.
- SIF Web Applications conforming to the architectural requirements defined in this specification are capable of transparently replacing any equivalent agent/application pair in the Zone ...and vice-versa.

2.1.2 The architecture will be extended by embracing the new technologies, not by replacing or patching existing ones.

Web service technologies such as SOAP and WSDL provide the framework for the second reference infrastructure for the SIF Standard.

- They co-exist with and do not replace the existing SIF infrastructure within a SIF Zone.
- Their usage is "logically consistent", and not simply a "wrapper" placed around the existing SIF infrastructure.

Components conforming to either infrastructure continue to be equal "citizens" in all SIF v2.5 and above Zones.

2.1.3 Incorporating a second reference transport within the SIF standard must not decrease "Out of the Box" application interoperability.

The set of requirements placed on the new transport is as proscriptive as those on the original. Where the functionality of an architectural style and a web service standard overlap (ex: REST and SOAP), or two versions of the same web service standard are incompatible to any degree (ex: WSDL 1.1 and WSDL 2.0) only one choice was made a normative dependency for the SIF standard.

In those cases where multiple incompatible options exist within the same version of a web service standard (ex: SOAP "literal" vs. SOAP "encoded") only one option was made a normative dependency for the SIF standard.

2.2 Glossary of Terms

The following terms are used throughout the rest of this document. Wherever possible, they reflect common industry usage and consensus.

2.2.1 Service

A Service is a software application that responds to requests made of it by client applications. Any given application can be both a service used by multiple clients, and a client which itself uses other services.

Every service possesses a public interface, defining exactly what operations its clients can ask it to do. This interface specifies the methods the service supports, the data these methods accept and the results they return. Each service also has a hidden (private) implementation which determines how it will actually "service" these requests.

The fact that the service implementation is hidden (encapsulated) means that even if the details of that implementation radically change, if the interface is unaffected, none of the clients of that service will be impacted. Having clients be independent of how a service is actually implemented is a key enabler of the architecture described in this document.

2.2.2 SIF Object Provider

A SIF Object Provider can be considered as an "Object Service" in the sense that it supports requests for object data from "clients" and responds by supplying that data. The full Service interface is identical for all Object Providers, and consists of Read object data (via the exchange of Request / Response messages) and Create, Update, and Delete object data (via publishing Event messages to subscribers) when changes occur to the data in any of its supported objects.

But any SIF Object Provider utilizing the HTTP/S platform is not a Web Service.

- It does not support the SOAP transport, and it does not have a defined WSDL
- Its interface encapsulates neither the cross-object relationships within the SIF object hierarchy nor the behavioral aspects of an educational process

Each of those limitations has been addressed … the first with the Web Service functionality described in this document, and the second with the addition of Zone Services in SIF v2.4.

2.2.3 SIF Zone Service

Starting in SIF v2.4, the existing SIF infrastructure was extended to support "non-CRUD" operations, which allowed Zone Services to be constructed which encapsulate both the details of the object hierarchy and associated transactional behavior (see Section 7).

Three new SIF message types (ServiceIn, ServiceOut and ServiceNotify) were required to carry the "non-object" operation requests, responses and event notifications respectively. As a result, any client of a Zone Service MUST support this extended SIF infrastructure.

All of the message types of the original SIF infrastructure (including these new Zone Service extensions) have been mapped to their SOAP equivalents.

2.2.4 Web Service

A Web Service is a Service that conforms to the following general requirements.

- The format of the data it accepts and produces is defined by XML Schema.
- Its defined operations (interface) are described by the Web Services Description Language (WSDL) and automatically generate "invocation stubs" in clients of that service.
- Each operation is "bound" to a specific XML schema which defines the contents of the associated data.
- Web clients interact with the web service in a manner prescribed by its WSDL description. They exchange data in formats defined by its bound XML schemas, carried over the SOAP transport in accordance with a set of conventions defined in the WS-I Basic Profiles (BPs) and Basic Security Profiles (BSPs).
- A family of additional WS-* standards provide the conformant Web Service with many of the messaging capabilities already provided by the Zone Integration Server (ZIS) for SIF applications within the Zone. These capabilities include reliable message delivery, content based routing, and automatic service discovery.

2.2.5 SIF Infrastructure Web Services (SIWS): v2.5 and above ZIS

The SIWS are a defined set of 5 web services, each represented by a WSDL file. Each of these WSDL files define a single Port Type (interface) and Port Binding, along with a set of defined operations and an implied choreography for operation invocation. All 5 interfaces and their contained operations MUST be provided and supported by any "web enabled" SIF v2.5 and above Zone Integration Server.

Taken together, these SIF Infrastructure Web Services provide their web clients with access to the complete range of existing ZIS functionality, although a given client may choose to invoke operations on only some of these SIWS. Any SIWS client must be capable of being a full participant in the SIF Zone, without relying on any specific characteristics of the ZIS implementation behind these interfaces.

2.2.6 SIF Application Web Client (SAWC): v2.5 and above SIF SOAP Agent

A SAWC is the web client of one or more of the SIF Infrastructure Web Services, and it must be able to invoke SIWS operations over the SOAP transport in a manner completely analogous to how a SIF agent/application invokes ZIS methods over the HTTP/S SIF transport.

If a SAWC replaces any existing SIF v2.x agent/application pair, such a substitution is (at the infrastructure level) invisible to any other components of the Zone, excluding the ZIS. This is true even if it replaces an Object Provider.

There are two varieties of SAWC, corresponding to SIF Pull Mode and Push Mode Agents.

Pull Mode
A SIF "Pull Mode" Agent using the SOAP transport is a pure web client, and uses SOAP only to invoke operations on the appropriate SIWS (ex: Register, Provision, Post Event, Query). It should (similar to a Pull Mode Agent on the HTTP/S transport) invoke the GetMessage operation of the appropriate SIWS to synchronously obtain the next asynchronous message from the delivery queue maintained for it by the ZIS.

Push Mode
A SIF "Push Mode" Agent using the SOAP transport also needs to be able to asynchronously receive incoming messages from the ZIS (ex: Query, QueryResponse, Event). This requires it to also support one or more web service interfaces with a range of operations that correspond to the expected types of those incoming messages.

2.2.7 SIF Application Web Services (SAWS): v2.5 and above SIF SOAP Push Mode Agent

In addition to being a SIF Application Web Client (SAWC), all Push Mode Agents over the SOAP platform MUST also support one or more of the 3 SIF Application Web Services, each defined by its corresponding WSDL file. Each WSDL file is composed of a single Port Type (interface) and Port Binding with a set of defined operations and an implied choreography for operation invocation.

A given Push Mode Agent indicates which among the 3 SAWS interfaces it will provide to clients, in the set of Property name / value pairs within the Protocol element it sends to the ZIS at Agent Registration time, where the "name" is the target namespace of the service WSDL (which includes the version number supported). For example the Property Name for the optional Data Model web service interface is:

http://www.sifassociation.org/contract/DataModel/2.x

The Push Agent MUST issue a valid response for every operation in each interface it provides, even if that response is an Error message.

It should be possible to construct a SOAP Push Mode Agent using any of a number of standard web service developer toolkits.

2.3 Architectural Components

Any SIF Application Web Client (Pull or Push Mode Agent) written to utilize the SOAP transport, and utilizing the set of SIF Infrastructure Web Service interfaces and implicit operation invocation choreography, must be able to

- Participate fully in the SIF Zone
- Interoperate seamlessly on an infrastructure level with the ZIS, other Web Application Clients and Services, and all agent/application pairs which utilize the original HTTP/S infrastructure.

This is illustrated in the following diagram, which is explained in further detail in the subsections below.

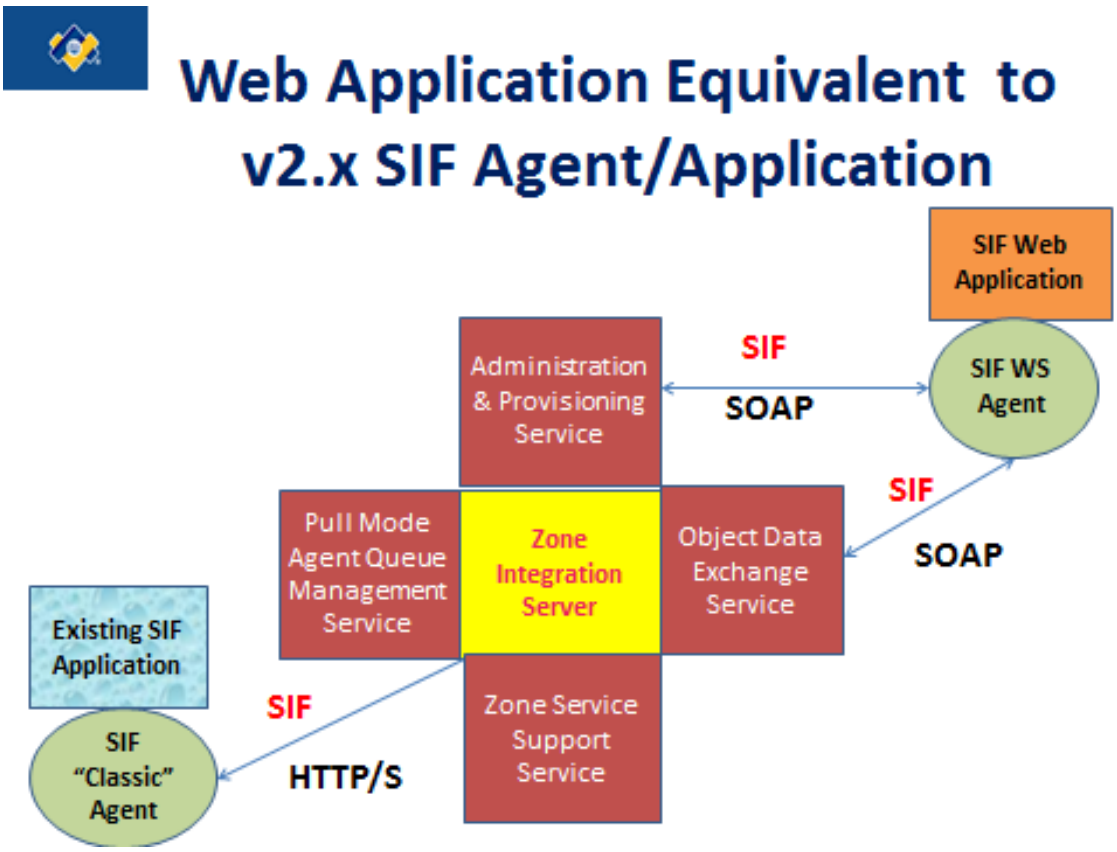


Figure 1: Example of the SIF Infrastructure Web Service (SIWS) seamlessly supporting exchanges between a SIF web application over SOAP, and a SIF agent/application component over the alternate HTTPS transport.

2.3.1 SAWC Required Capabilities

The functions provided by the HTTP/S SIF agent are also necessary for the SIF Web Application Client. It should contain logic which:

- Invokes the SIF Infrastructure Web Service WSDL operations over the SOAP transport.
- Uses the SIF XML standard message schema to invoke a SIWS operation and interpret its return.
- Operates in conformance with the existing agent/ZIS choreography.

The basic set of capabilities provided to a SIF Application Web Client / Web Service by the SIWS must allow it to:

- Register and Provision itself in the Zone
- Get information about the other applications previously registered in the Zone
- Request SIF object data and receive valid (and understandable) Responses from the Object Provider for the object type selected.
- Subscribe to and receive Events for one or more SIF object types.
- Serve as the Object Provider for one or more object types. This includes receiving all posted Requests for object data and having all provided Responses routed back to the correct issuing client.

- Publish Events (whether or not the SAWC / SAWS is serving as the Object Provider for that object type).
- Function as a Zone Service, receiving ServiceInput and issuing ServiceOutput and ServiceNotify messages in response.
- Function as a client of a Zone Service, issuing ServiceInput and receiving ServiceOutput and ServiceNotify messages in response.
- Support the existing SIF message packetization functionality for those message types where it is required.
- Optionally support Directed Request, Directed Event, ServiceInput and ServiceNotify messages to a specified recipient.
- Function even though it cannot receive incoming HTTP/S connections (i.e. is operating in Pull Mode).

2.3.2 ZIS Optional Capabilities

Support for the following capabilities MAY be provided, but is not required of the ZIS.

> **Interoperability with agent/applications from earlier versions:** Backward compatibility can optionally be extended to include interoperability (on an infrastructure level) with SIF v1.5 components and earlier.

2.3.3 Migration Requirements

Any v2.x deployed SIF Zone can be made "web enabled" by upgrading the ZIS to a version (v2.5 or later) that supports the SOAP/WSDL mapping described in this document.

No other agent/application component changes need be made. Everything interoperates exactly as before. The difference is that SIF Push and Pull Agents can now register, and freely interoperate within the Zone over SOAP using the SIWS WSDL, because the SIF v2.6 ZIS MUST support both reference transports.

3 Web Service Framework

The following set of web standards, versions and options are used by SIF Web Applications to exchange XML documents with the ZIS. The collection of these normative dependencies is referred to as the Web Service Framework. All ZIS and SAWC/SAWS MUST support this framework.

Component	Choice	Options
Transport	SOAP 1.1	Document/Literal
Interface Language	WSDL 1.1	
Basic Profile	BP 1.2	WS-Addressing
Security	Transport Layer Security (TLS) Underlying protocol: HTTPS 1.1	Mutual Authentication mode supported by X.509 Certificates

The specific reasoning behind the selection of each of these individual components is described in an associated document called “SIF Web Services: Decision Context”, contained within the SIF Web Services Developer Release Kit.

4 SIF HTTP/S Infrastructure to SOAP Mapping

Applications supporting two dissimilar transports can be made to interoperate by inserting a "transport bridge" between them, which bi-directionally intercepts each message and forwards it along over the new transport to the intended recipient.

The ZIS is the intermediary target of every message sent between SIF applications, and MUST bi-directionally map between the existing SIF HTTP/S infrastructure elements and the SOAP header/body message parts. This enables web applications utilizing SOAP to transparently exchange SIF object data with previously certified SIF applications - with neither side being aware of the intermediate ZIS-provided Bridge.

All SIF messages sent over the SOAP transport are divided into a SOAP Header and SOAP Body. The SOAP Header traditionally provides a defined location in the message structure for communicating Quality of Service (QoS) specifics such as reliability, addressability, and security as well as custom specifications. The SOAP Header transporting a SIF message has:

- A normative dependency on the WS-Addressing standard. It contains the required set of wsa: elements (wsa:To, wsa:From, etc.).

- A relocated SIF_Header element (SIFHeader) which provides additional "routing type" information that was formerly located within a SIF message schema (such as *SIF_Event@ObjectName*).

The SOAP Body contains a single SIF message, of a type and form which is completely specified by the SOAP Header and the corresponding SIF message schema for the SOAP transport (such as *Event*).

The following sections detail the complete SOAP Transport mapping.

4.1 New set of SIF Message Schema equivalents for the SOAP Transport

A completely parallel hierarchy to the HTTP/HTTPS SIF_Message subschemas (ex: SIF_Event, SIF_Register) is defined for messages being transported over SOAP. Each subschema appears in modified form in its SOAP equivalent (with the "all the "SIF_" prefixes removed). This was done for the following reasons:

- Information in the SOAP Header describes what type of message is being conveyed, and how to route it. The SOAP Body contains the specific operation input or output data. The SOAP Envelope therefore replaces the single top level SIF_Message container, which has been eliminated.

- Elements related to the routing or interpretation of a SIF message (including but not limited to those contained in complex elements like SIF_Response/SIF_Header and any associated packet control) have been moved out of the individual message subschemas into the aforementioned SIF complex element (SIFHeader) located within the SOAP Header.

  While this header information will not always prevent the ZIS from needing to examine the SOAP Body before routing the SOAP message (for example when supporting XML Filtering requirements, or where payload version conversion needs to be performed), it SHOULD eliminate that necessity in the vast

majority of remaining cases. It also provides a much more "SOAP-natural" mapping for the data being exchanged.

4.1.1 Separation of Data Model and SOAP Messaging infrastructure

The set of WSDL files and their SOAP binding (the accompanying Transport and Message schemas) are decoupled from (independent of) the SIF Data Model. This means the SOAP infrastructure can be used without change to support multiple locales (US, UK, AU), State-specific Data Profiles or new versions of the Data Model itself (new objects and / or elements).

4.1.1.1 High Level Changes

This required the following high level changes to the previous hierarchy of SIF_Message Schemas.

High Level Schema Change	Details
Infrastructure Data Objects	ZoneStatus and AgentACL are the two objects which belong to infrastructure, and (as is true for the HTTP/S transport) are specifically requested and returned through separate message types rather than Request / Response sequences. Their Metadata element is pruned of Data Model-specific subelements such as EducationFilter.
Data Object "Schema Choices"	The elements which provide a "choice" spanning multiple members of the set of Data Object schemas (ObjectData, EventObject, ReportDataObject and ResponseObjects) are redefined to be of BodyType. BodyType is an "xs:any", originally created to decouple the Zone Service operation schemas from the specific Zone Service data schemas, and LogEntry from the schema of the object being logged. It plays the same role for Web Service operations. Information *about* the Event Object, Report Data Object and Response Object is contained in the SOAP Header, while the appropriate "xs:any" element itself is carried in the SOAP Body. This change means adding or deleting elements or object types from the SIF Data Model will not impact existing SOAP message schemas. Only the content of the SOAP Body on the wire will be changed.
Data Object and Service "Name Enumerations"	The elements which provide an enumerated list of SIF Object or Service names (Provide/ObjectNames, Request/ObjectNames, Subscribe/ObjectNames, LogObjectName, ServiceInput/Service, and ServiceNotify/Service) are redefined to TopicName (an unqualified XML token). This change means adding or deleting object types in the Data Model will not impact any enumerated list of such types within the SOAP message schemas.
SIFHeader	The SIF_Header element found in all SIF HTTP/HTTPS messages has been expanded and moved to the SOAP Header. It encapsulates all SIF-related elements needed to understand how to process and route a SIF message. In

	every case the information it contains is NOT replicated within the SOAP Body.
"SIF_" Prefix	No SIFHeader subelement in the SOAP Header has a "SIF_" prefix (so for example the "SIF_Timestamp" element is conveyed as "Timestamp"). No SIF message (non-Data Model) element appearing in the SOAP Body has a "SIF_" prefix. For example, the SOAP transport equivalent to a SIF_Event message is posted by issuing a WSDL "Event" operation with the "Event" element as the top child of the SOAP Body. The Data Model elements (all those specific to the object being conveyed) are identical to those elements within the SIF_Message namespace, including whether or not a "SIF_" prefix appears in their element names.
Zone Service Messages	The SOAP Body directly carries the SIF_Body subelement (an xs:Any) of the ServiceInput, ServiceOutput or ServiceNotify operations. All other subelements of the HTTP/S SIF_Service message schemas (SIF_ServiceMsgId and SIF_Operation, along with elements SIF_MaxBufferSize, SIF_Error and SIF_Version) appear in SIFHeader.
Other	"HTTP-SOAP1.1 and "HTTPS-SOAP1.1" are added as enumerated values in DefinedProtocolType.

4.1.1.2 Message Type-specific Changes

The following illustration contrasts the top level element mapping between a SIF Response message being sent across each of the supported transports.

HTTPS Transport	**SOAP Transport**
Message SIF_Response SIF_Header Header Sublements Packet control elements SIF_ObjectData (xs:choice) (Any Data Object Schema)	SOAP Envelope SOAP Header WS-Addressing elements SIFHeader Header Subelements Packet control elements SOAP Body QueryResponse (xs:any) (Any Data Object Schema)

The SOAP Transport example reflects the following level of decoupling:

- SIF_Response has been broken up into two possible SOAP Messages: QueryResponse (shown here) and DataModelError (elevating its SIF_Error subelement to its own message type).
- The SIFHeader contains those elements necessary to successfully process and route the QueryResponse message being conveyed
- The SOAP Body contains the QueryResponse schema as its top child element
- The QueryResponse schema is independent of the actual SIF data object and its subelements.

4.1.2 Namespaces

There are three namespaces which define the SIF contents of every SIF SOAP message. Each is independent of the others.

Data Model (and Objects): http://www.sifinfo.org/infrastructure/2.x
Identical to the single SIF_Message XML Namespace supported from SIF v2.0 r1 on, which encapsulates both the HTTP/S infrastructure and the Data Model being released. Those subelements of this namespace which define the Data Model, and are independent of the infrastructure, are reused and contained within the SOAP Body. An example of a top level element from this namespace contained within the SOAP Body is "GradingAssignment".

Messaging: http://www.sifassociation.org/message/soap/2.x
The XML Namespace which scopes the SIF message schemas that wrap the data objects carried within the SOAP Body. It replaces the high level messaging elements in the Data Model namespace, and is independent of both the Data Model schemas and the SOAP Transport schema.

No element has a "SIF_" prefix. An example of a top level element from this namespace is "ExtendedQueryResponse"

Transport: http://www.sifassociation.org/transport/soap/2.x
The XML Namespace which scopes the SIF elements contained within the SOAP Header. It is specific to the SOAP 1.1 transport and is independent of both of the other namespaces.

No element has a "SIF_" prefix. The single top level element of this namespace is "SIFHeader", and it is one of the top level elements in the SOAP Header.

4.2 SIFHeader Elements within the SIF Message SOAP Header

The SOAP Header for a SIF message contains the set of elements mandated by the WS-Addressing standard, and the complex element "SIFHeader". Coupled with the WS-Addressing elements, the SIFHeader subelements provide all information either required by the ZIS to route a SIF message, or by a SIF application to understand the message that has arrived.

SifHeader subelements are either general (common to all SIF messages on the SOAP transport) or they contain information specific to one or more message types (ex: "Query") and are conditionally present only in the SIF Header of a message of that type..

4.2.1 General Messaging Information

Each of the following elements can be optionally contained in the SOAP Header part of every SOAP message being exchanged between a ZIS and a SOAP agent, and some are required to be present in every such message. Where equivalent subelements in this section exist within the original SIF_Message schema hierarchy they are removed in their SOAP Transport equivalents.

Note: Any elements identified after the phrase "equivalent to" in the tables below refer to the original HTTPS SIF Message schemas. Any elements with characteristics flagged as "WSA" indicate they are provided in the SOAP transport by WS-Addressing.

SIFHeader subelements contained in the SOAP Header	**Char**	**Usage/Meaning/Equivalent in HTTPS Transport Message Schema**
InfrastructureVersion	M	Version of the SOAP Infrastructure (both the Transport and Messaging Namespaces) are conformant with. It is a string value restricted to the format "#.#r#" where the "r#" is optional (ex: 2.0r1). Set to "2.6" for this release.
DataModel	C	Data Model identifier which defines the schema for the set of XML elements contained in the SOAP Body payload of this message. For the US v2.6 release it is a URI with a value of: http://www.sifinfo.org/infrastructure/2.x This element is mandatory for messages with Object data contained in the SOAP Body, and optional otherwise.
DataModelVersion	C	Version of the actual Data Model being carried in the SOAP Body. It is equivalent to the SIF_Version attribute contained in SIF_Message. It is a string value restricted to the format "#.#r#". Set to "2.6" for this release This element is mandatory for messages with Object data contained in the SOAP Body, and optional otherwise.
ZoneId	C	URI which uniquely identifies the Zone containing both the Sender and Receiver of this SOAP message. It has the form: • urn:sif:zone:xxx.yyy.zzz where xxx.yyy.zzz is a structure that reading left to right starts with most specific identification such as school and works rightward to identify the higher levels. (*urn:sif:zone:AcmeMiddleSchool1.CoyoteDistrict.Arizona*) The initial (most specific) field after "zone" SHOULD be identical to the SIF_ZoneId attribute returned in SIF_ZoneStatus (Ex: *AcmeMiddleSchool1*) This element is mandatory for all messages except the initial Registration by the agent (Push or Pull). It MUST be returned by the ZIS in the Status message following a successful Registration, and is included as an attribute in the ZoneStatus object.

Security	O	Equivalent to the SIF_Security element in the HTTP/S Transport SIF_Header. Because it is in the SOAP transport all "SIF_" prefixes have been removed from its member elements as well. A complex element which allows an originating agent to specify security requirements that the ZIS must ensure upon delivery of the message to recipient agents.
Timestamp	M	Equivalent to the SIF_Timestamp element in the HTTP/S Transport SIF_Header Time of message creation.
Message Id	WSA	The SIF_MsgId element in the HTTP/S Transport SIF_Header is conveyed in the wsa:MessageId element in the SOAP Header. The value of that element is unique for every SIF message sent. While this element is optional in the WS-Addressing standard, it is required in the SOAP Header for all SIF Messages. The developer must specify the element as an RFC 4122 compliant UUID with the "urn:uuid" prefix and the type (13th Digit) set to either a 1 (MAC Address) or 4 (Random). In this way it can be directly converted into a SIF MessageId value, and back. For example, a SIF Message ID of: "12345678123**4**567123456781234 5ABC" Becomes the WS-Addressing Message ID equivalent of: "urn:uuid:12345678-1234-**4**567-1234-567812345ABC"
Message Name	WSA	This element which has no direct counterpart in the SIF_Message hierarchy is equivalent to an enumerated list which corresponds to the set of all possible SIF messages contained in the SOAP Body. Its value is conveyed in the wsa:Action element in the SOAP Header, and is unique for every SIF message type.
SourceId	M	Equivalent to the SIF_SourceId element in the HTTP/S Transport SIF_Header. It is RECOMMENDED that the ZIS set this value to the most "specific" field in the ZoneId attribute returned in the ZoneStatusResponse message (assuming the *URN* alternative form for that value was used). For Soap agents, it is equivalent to the value placed in the SourceId of every message they send, which must match up with the SourceId value they provided in the Register Message.
DestinationId	O	Equivalent to and follows the rules for SIF_DestinationId in the HTTP/S Transport SIF_Header. It is used by the ZIS to content-route the message based upon matching it with a pre-stored SourceId. For Soap agents it is set only if the message is a "Directed Event" or "Directed Request" where the client instead of the ZIS determines the actual recipient.

Contexts	O	Equivalent to the SIF_Contexts element in the HTTP/S Transport SIF_Header. Because it is in the SOAP transport all "SIF_" prefixes have been removed from its member elements as well. The list of Contexts to which the message applies. Currently only the default value is officially defined.
PacketData	O	Complex element present whenever the SOAP message is a packet in a larger SIF Message. It allows the SOAP transport to assign a unique Message ID to each packet, while providing enough additional information to allow complete reconstruction of the Message when bridging back to the HTTPS transport.
PacketData/PacketNumber	M	xs:positive integer corresponding to the packet number
PacketData/MorePackets	M	xs:token with value of YES or NO

The following XML instance fragment shows the part of the SIFHeader within the SOAP Header, for an Event Operation being invoked on the SIF Student Admin Application Web Service in the Acme Middle School Zone.

```
<SIFHeader xmlns="http://www.sifassociation.org/transport/soap/2.x" soap:mustUnderstand="1">
    <Timestamp>2010-10-24T15:58:33.984Z</Timestamp>
    <InfrastructureVersion>2.6</InfrastructureVersion>
    <DataModel>http://www.sifinfo.org/infrastructure/2.x</DataModel>
    <DataModelVersion>2.6</DataModelVersion>
    <ZoneId> urn:sif:zone:AcmeMiddleSchool1.CoyoteDistrict.Arizona.US</ZoneId>
    <Security>
        <SecureChannel>
            <AuthenticationLevel>3</AuthenticationLevel>
            <EncryptionLevel>4</EncryptionLevel>
        </SecureChannel>
    </Security>
   <SourceId>RamseySIS</SourceId>
   <DestinationId>StudentAdmin</DestinationId>
```

4.2.2 Message Type-Specific Information

The SIF Message namespace consists of message-specific elements taken from the single HTTP/S namespace, which are unrelated to either the Data Model or the SOAP transport. They are organized as a collection of WSDL operation (message type) schemas each of which defines the top level element in the SOAP Body conveying that operation.

The following table lists the source and Message Namespace mapping of these operations.

SIF_Message Component -with equivalent WSDL operation(s) in "(…)"	SIF Message namespace mapping
SIF_Message	SIF_Message contains the release namespace and version, and, as a child element, every SIF Message. It has been removed in the SOAP mapping. The lower level message elements form the payload of every SOAP Body.

	The "Namespace" value is carried in the SIFHeader element "DataModel". The "Version" is carried in the SIFHeader element "DataModelVersion.
SIF_Ack SIF_Header SIF_OriginalMsgId and a choice of either: SIF_Status SIF_Error	SIF_Ack is the synchronous HTTP/S transport response to every issued message. It spans the cases where either a SIF_Status (success) or SIF_Error (Error) is being returned. All the elements in SIF_Header are mapped to SIFHeader subelements in the SOAP Header. OriginalMessageId is conveyed in the WS-Addressing element wsa:RelatesTo in the SOAP Header.. As a result, Status and DataModelError are both top level WSDL operations (see below) and one or the other is returned in place of SIF_ACK as the top level element in the SOAP Body.
SIF_Error (SOAP Fault or DataModelError)	SIF_Error spans the cases where an immediate (synchronous) transport error for an operation just issued is being reported (SOAP Fault), or an asynchronous error occurring at a higher level is being reported either: • In response to an earlier SIF_Request or SIF_ServiceInput operation. • As the "one-way" Pull Mode Agent's asynchronous response to a SOAP Envelope message received in response to an earlier GetMessage. • As the next Packet in a packetized message when an error on the sender side makes it impossible to continue In the case of an asynchronous higher level error, the Error element (renamed "DataModelError"), is returned contained in the SOAP Body. It has the identical structure to SIF_Error, with the "SIF_" prefix removed: DataModelError Category Code Desc ExtendedDesc Where such an error is being issued in response to a Zone Status operation, the SIFHeader will contain the conditional OriginalMessageType element, which will indicate what type of message generated the error (either "ServiceIn", "ServiceOut", "ServiceNotify", or for completeness "Request"). In the case of a synchronous transport error, the Error subelements are mapped to equivalent subelements in the SOAP v1.1 Fault element. The SIF WSDL specifies that SOAP Fault is to be returned synchronously (on the same HTTP/S connection) in the SOAP Body, as the **fault** part of every SIF WSDL operation whenever a synchronous transport level error occurs.
SIF_Status SIF_Code [M SIF_Desc [O] SIF_Data [O]	SIF_Status spans three different message choreographies: 1. The synchronous response to an earlier "Get" operation is being returned. There is an internal SIF_Data subelement containing the corresponding response.

and one of the following choices if Data is being returned: SIF_Message [C] SIF_AgentACL [C] SIF_ZoneStatus [C]	2. The synchronous transport status of any operation other than a just issued "Get" or "Register" message is being reported. There is no internal SIF_Data subelement. 3. An asynchronous Status invocation on the ZIS is issued by a Pull Mode Agent. It reports the successful arrival of the synchronous Status response to an earlier SIF_GetMessage operation (allowing the ZIS to delete the message from the Pull Agent's queue). There is no internal SIF_Data subelement in this case. Each is expanded below.
SIF_Status with SIF_Data (AgentACLResponse, ZoneStatusResponse, or GetMessageResponse)	The SIF_Status with SIF_Data is removed because SIF_Code and SIF_Desc are redundant, as illustrated below. • When SIF_Data contains an AgentAcl or ZoneStatus, it indicates the "Get" operation was a success. In this case the SOAP response is equivalent to a SIF_Code of 0, with no SIF_Desc. • When SIF_Data contains a Message, the SOAP response is likewise equivalent to a SIF_Code of 0, with no SIF_Desc. The Message is represented in the SOAP Body by a complete SOAP Envelope. SIF_Data (now empty except for its choice of subelements) has also been removed. One of its three subelements is now the **output** returned synchronously (via the same HTTP/S connection) upon the success of the **input** part of the corresponding WSDL "Get" operation. This is the only time a Status message is not returned. Specifically: • AgentACL (less SIFHeader information) is returned in the AgentACLResponse message in response to GetAgentACL or Register. • ZoneStatus (less SIFHeader information) is synchronously returned in the ZoneStatusResponse message in response to GetZoneStatus. • SOAP-ENV containing the SOAP Header and SOAP Body of the next message for a Pull Mode Agent (the equivalent of SIF_Message), is synchronously returned in the SOAP Body in response to that agent issuing a GetMessage.
SIF_Status without SIF_Data (Status)	The Status message itself is returned in the SOAP Body. It includes Code and Desc, and there is no SIF_Data. This is the top level SOAP Body element for the **output** "part" of almost every SIF WSDL operation, sent whenever the **input** part was successfully received. 4. It is also sent asynchronously by a Pull Mode Agent to indicate successful reception of a SOAP_Env response to a GetMessage, allowing the ZIS to remove the message from the Pull Agent's queue. As is true with the HTTP/S agent, the ZIS must respond with a synchronous Status message, indicating whether (0) or not (9) there are additional messages in the queue.
SIF_Event SIF_Header	The SIF_Header information in the Event is mapped to SIFHeader.

SIF_ObjectData SIF_EventObject @Object Name @Action Object Data	The ObjectName attribute is mapped to the TopicName element in SIFHeader, but as a non-qualified token rather than an enumerated list. This keeps the SOAP Header independent of the Data Model details for the messages being conveyed. The Action attribute ("Add", "Change" or "Delete") is mapped to the "EventAction element in the SIFHeader. The only information remaining under SIF_Event is the actual object data in the Event (an xs:any) which is carried in the SOAP Body under the Event element name.
SIF_Request SIF_Header SIF_Version SIF_MaxBufferSize and either: SIF_Query SIF_ExtendedQuery	As is true in every case, SIF_Header information is mapped to SIFHeader. SIF_Request has been removed. Query and Extended Query are the new top level elements. SIF_Max BufferSize is mapped to the SIFHeader element "MaxBufferSize" and SIF_Version is mapped to the SIFHeader element "ResponseVersion". Query and ExtendedQuery are each a unique WSDL operations.
SIF_Response SIF_Header SIF_RequestMsgId SIF_PacketNumber SIF_MorePackets SIF_MaxBufferSize and either: SIF_ObjectData SIF_ExtendedQueryResults SIF_Error	As is true in every case, SIF_Header information is mapped to SIFHeader. SIF_RequestMsgId is conveyed in the WS-Addressing element wsa:RelatesTo in the SOAP Header. As is true in every case where packet information occurs within a message, it is mapped to subelements of "PacketData" in SIFHeader. SIF_MaxBufferSize is mapped to SIFHeader element "MaxBufferSize". SIF_Response has been removed. Each of the three choices are a unique WSDL operation: • QueryResults (formerly SIF_ObjectData) • ExtendedQueryResults • DataModelError
SIF_SystemControl	SIF_SystemControl consists of a SIF_Header element and a choice of one of eight messages. As is true in every case, SIF_Header information is mapped to SIFHeader. Each of the eight choices is a unique WSDL operation (Ping, Sleep, Wakeup, GetMessage, GetZoneStatus, GetAgentACL, CancelRequests, CancelServiceInputs)
Zone Status Operations SIF_ServiceXXX SIF_Header SIF_Service SIF_Operation SIF_ServiceMsgId SIF_Version SIF_MaxBufferSize SIF_PacketNumber	As is true in every case, SIF_Header information is mapped to SIFHeader. SIF_Service is stored in "TopicName" in SIFHeader. SIF_Operation and SIF_ServiceMsgId are mapped to a subelements of the conditional element "ZoneServiceData" within SIFHeader. On the HTTP/S transport, SIF_Service and SIF_Operation are enumerated lists of names tied into a specific set of Zone Services. Their SOAP transport equivalents are non-qualified tokens so that the SOAP transport remains independent of the set of Zone Services it is supporting.

SIF_MorePackets and a choice of: SIF_Body SIF_Error	The value of SIF_Version is contained in the DataModelVersion element within SIFHeader. SIF_MaxBufferSize is contained within SIFHeader. As is true in every case where packet information occurs within a message, it is mapped to subelements of "PacketData" in SIFHeader. The SOAP Body for all successful Zone Service operations will consist of a single child element (ServiceInput, ServiceOutput or ServiceNotify) whose contents are equivalent to SIF_Body (xs:Any). Whenever an error is reported (either an illegal ServiceIn operation was made or a packet error occurred), the SOAP Body will contain a ZoneServiceError This allows any Zone Service message to be carried by the SOAP transport.

Example:
The following XML instance fragment shows the entire SIFHeader within a SOAP Header, for a Change Event Operation on the StudentPersonal Object, being invoked on the Student Admin SIF Web Application in the Acme Middle School Zone.

```
<SIFHeader xmlns="http://www.sifassociation.org/transport/soap/2.x" soap:mustUnderstand="1">
  <Timestamp>2010-10-24T15:58:33.984Z</Timestamp>
  <ZoneId> urn:sif:zone:AcmeMiddleSchool1.CoyoteDistrict.Arizona.US </ZoneId>
  <InfrastructureVersion>2.6</InfrastructureVersion>
  <DataModel> http://www.sifinfo.org/infrastructure/2.x </DataModel>
  <DataModelVersion>2.6</DataModelVersion>
  <Security>
      <SecureChannel>
          <AuthenticationLevel>3</AuthenticationLevel>
          <EncryptionLevel>4</EncryptionLevel>
      </SecureChannel>
  </Security>
  <SourceId>RamseySIS</SourceId>
  <DestinationId>StudentAdmin</DestinationId>
  <TopicName>StudentPersonal</TopicName>
  <EventAction>Change</EventAction>
</SIFHeader>
```

4.3 WS-Addressing Elements within the SIF Message SOAP Header

All SIF SOAP messages MUST support WS-Addressing. This is not reflected in the supplied WSDL, as explicitly mandating such support demands inclusion of WS-Policy assertions that may not be supported on every platform. These assertions are not required since WS-Addressing support is implicit in all SIF SOAP messages, and does not need to be optionally declared in the WSDL Port Type binding.

Support for including WS-Addressing elements in the SOAP Header must therefore be provided by the Web Service development platform. Wherever a WS-Addressing element is needed, it is mandatory in that situation,

even if the WS-Addressing standard considers the element to be optional. The SIFHeader does not duplicate WS-Addressing SOAP Header elements.

The following table indicates how the values for the WS-Addressing elements are assigned. Whenever an End Point Reference (EPR) is indicated, only its "address" subelement is mandatory.

WS-Addressing Element	Char	Usage in SIF Message SOAP Header
wsa:To	C	URL of destination. This element is required when the issuer of a SIF message initiates an HTTP/S connection. It MUST be omitted (or set to the WS-Addressing "anonymous" setting of *http://www.w3.org/2005/08/addressing/anonymous)* when a Status or Error response is sent back synchronously over the same connection. For all agent initiated messages exchanges this MUST be set to the ZIS URL. For all ZIS initiated message exchanges it MUST be the URL provided by the PushMode Agent at Registration time (contained within the matching Protocol/Property subelement in the Register message).
wsa:From	O	Endpoint Reference of Source. The "Address" subelement within the From EndpointReference is determined as follows: **1. All ZIS issued messages** The URL for the ZIS (where the initial Agent Register message was sent). **2. An Agent is issuing a client Request to the ZIS.** For a PushMode Agent, the value is the URL the agent provides at Register time (contained within the Protocol subelement in the Register message). For a Pull Mode Agent, the wsa:From element is not required **3. A Push Mode Agent is responding to a ZIS-initiated operation.** In this case the URL provided by the agent will match the URL that the Service operation was invoked on.
wsa:MessageID	O	The value is unique for all messages in the Zone, and it replaces SIF_MsgId. As discussed, it is an RFC 4122 compliant UUID, with the 32 hex digit SIF Message ID contained in the string "urn:uuid:{8 digits}-{4 digits}-{4 digits}-{4 digits}-{12 digits}" with the 13th digit set to a 4 (random) or a 1 (MAC Address) as in the example "urn:uuid:12345678-1234-**4**567-1234-567812345ABC"
wsa:Action	M	This defines the SOAP Body Contents, and indicates the operation being requested. It supersedes the value of the soapAction HTTP attribute. If that value is present, it MUST agree with the value in wsa:Action. The rules for constructing this AttributedURI are taken from the WS-Addressing standard conventions for WSDL1.1. The names of each input and output operation are included directly in the SIF WSDL. This requires that the Action values MUST be:

		Input: {target namespace}/{port type name}/{input name} Output: {target namespace}/{port type name}/{output name} Fault: {target namespace}/{port type name}/{operation name}/Fault/{fault name} Example: Provision message from agent to ZIS <In>: **http://www.sifassociation.org/contract/Administrate_Provision-S11/2.x/ ptAdministrate_Provision/ProvisionRequest** Event message from ZIS to agent **http://www.sifassociation.org/contract/DataModel/2.x/ ptDataModel/PostEvent** Soap Fault response from ZIS to agent <Fault>: **http://www.sifassociation.org/contract/SifZone/2.x/ Zone-S11/Provision/Fault/ProvisionError** Because of the asynchronous nature of SIF data exchanges, except for "Get" requests and the initial Register, the Output part of every SIF WSDL operation is a Status (with status code), and the Fault part is a SOAP Fault. For example, a QueryResponse message is sent as the Input part of a new WSDL operation (triggered by the earlier reception of a Query message), rather than as the Output part of the Query operation. The same is true for the relationship between ServiceInput and ServiceOutput. It is also possible that a SOAP Fault may be sent for transport or encoding errors. For these situations, the value of wsa:Action is predefined as: http://www.w3.org/2005/08/addressing/soap/fault A WS-Addressing Fault may also be returned in the case where the destination is unreachable or where the requested service action was not supported.
wsa:RelatesTo	C	The value and relationship type of a related Message Id. Its element value replaces the one of the HTTP/S transport Message Id values in the following cases: • SIF_OriginalMsgId in SIF_Ack (used in Status and Error messages) • SIF_RequestMsgId in SIF_Response (Used in Query and Extended Query Response messages) • SIF_ServiceMsgId in SIF_ServiceOutput Its "RelationshipType" attribute MUST have a value of: http://www.w3.org/2005/08/addressing/reply which is the standard WS-Addressing pattern indicating the relationship between this message and the Message Id value in wsa:RelatesTo is that of the single Request / single Response MEP exchange.

wsa:ReplyTo	O	This is the recipient of the Status message for a successful operation. It can be omitted, but if present its address subelement MUST have the value "http://www.w3.org/2005/08/addressing/anonymous" This is the "anonymous URI" which indicates that there is no real endpoint available for this address. Therefore any reply must be sent back in the HTTP/S response message (i.e. synchronously). Any other value would cause the HTTP/S connection to be broken and a new one established to the URI specified here (i.e. an asynchronous message pattern).
wsa:FaultTo	O	This is the recipient of the Error message for a failed operation. It can be omitted, but if present its address subelement MUST have the value "http://www.w3.org/2005/08/addressing/anonymous" This is the "anonymous URI" value which indicates that there is no real endpoint available for this address. Therefore any error will be sent back in the HTTP/S response message (i.e. synchronously). Any other value would cause the HTTP/S connection to be broken and a new one established to the URI specified here (i.e. an asynchronous message pattern).

4.4 Illustrated Example

The complete XML instance of the SOAP Message conveying a StudentPersonal Change Event (update phone number) to the Student Admin SIF Web Application in the Acme Middle School Zone (from the ZIS) is shown below. The WSDL operation is "Event".

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope">
  <soap:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:To>https://AcmeHost:443/StudentAdmin</wsa:To>
    <wsa:From>
      <wsa:Address>https://AcmeHost:443/ZoneControl</wsa:Address>
    </wsa:From>
    <wsa:MessageID>urn:uuid:12345678-1234-4567-1234-567812345ABC</wsa:MessageID>
    <wsa:Action>http://www.sifassociation.org/contract/DataModel-S11/2.x/ptDataModel/PostEvent</wsa:Action>
   <wsa:ReplyTo>
      <wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
   </wsa:ReplyTo>
   <wsa:FaultTo>
      <wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
   </wsa:FaultTo>

   <SIFHeader xmlns="http://www.sifassociation.org/transport/soap/2.x" soap:mustUnderstand="1">
     <Timestamp>2010-10-24T15:58:33.984Z</Timestamp>
     <ZoneId>urn:sif:zone:AcmeMiddleSchool1.CoyoteDistrict.Arizona.US</ZoneId>
     <InfrastructureVersion>2.6</InfrastructureVersion>
     <DataModel> http://www.sifinfo.org/infrastructure/2.x </DataModel>
     <DataModelVersion>2.6</DataModelVersion
     <Security>
          <SecureChannel>
               <AuthenticationLevel>3</AuthenticationLevel>
               <EncryptionLevel>4</EncryptionLevel>
          </SecureChannel>
     </Security>
     <SourceId>RamseySIS</SourceId>
     <DestinationId>StudentAdmin</DestinationId>
     <TopicName>StudentPersonal</TopicName>
     <EventAction>Change</EventAction>
   </SIFHeader>
</soap:Header>

<soap:Body>
 <Event xmlns="http://www.sifassociation.org/message/soap/2.x">
     <StudentPersonal xmlns=" http://www.sifinfo.org/infrastructure/2.x"
          RefId="D3E34B359D75141A8C3D00AA001A1652">
        <PhoneNumberList>
          <PhoneNumber Type="0096"> <Number>(312) 555-1234</Number> </PhoneNumber>
        </PhoneNumberList>
     </StudentPersonal>
 </Event>
</soap:Body>

</soap:Envelope>
```

Notes

- The SOAP Header consists of global WS-Addressing elements and a SIFHeader complex element. The SIFHeader element "must be understood", so if a non-SIF application receives a SIF SOAP message, it will throw a SOAP Fault. None of its subelements have "SIF_" prefixes

- The value of wsa:To used by the ZIS matches the URL contained in the DataModel Property of the Protocol element in the original Register message sent by the receiving agent.

- The value of wsa:action will be the same for all SIF v2.6 SOAP Events, no matter in what Zone that event message is issued. It is a URL consisting of "namespace/port type/input name" where port type is "ptDataModel" and input is "PostEvent". This will be true for all other message types as well.

- The SOAP Body of this SIF message contains the single "Event" child element from the Message namespace which is of type xs:Any. All of its subelements are defined by the Data Model namespace schema of the SIF object being conveyed.

4.5 Transport Errors

The SOAP transport has a different set of errors than the SIF_ACK/SIF_Error mechanism, and a completely different way of representing them (for example strings instead of numeric codes are used).WS-Addressing then maps these SOAP faults into the SOAP Body in a very specific way.

This subsection defines the error message/error code mapping between the two transports so that error conditions can be reported and understood between SOAP and HTTPS agent/application pairs,

4.5.1 SOAP 1.1 Faults

SOAP errors are conveyed through "SOAP Faults". These messages always have the following wsa:Action value:

<wsa:Action>http://schemas.xmlsoap.org/ws/2004/08/addressing/fault<wsa:Action>

The SOAP Body contains a top level SOAP Fault element consisting of a set of subelements, only the following of which are important to the SIF error mapping:

- Faultcode: Only one of 4 possible values (see table below)
- Faultstring: A description of the problem (always equated to Error/Desc)
- Detail: Application specific error information which must be included if the fault was caused by the contents of the SOAP Body of the message being reported, but must not be included otherwise.

There are 4 "categories of SOAP v1.1 errors. Each is described in the table below. Note that conversion to SIF Error category and Error Code is implicit in 3 of them.

The "behavior" of the SOAP Fault (including whether the recipient should resend the message generating the error) should be based on the equivalent HTTP/S transport behavior in all cases.

SOAP Error Category	Meaning	SIF Equivalent	Try Again?
VersionMismatch	Invalid Namespace in SOAP Element (ex: SOAP v1.2 message received)	Faultstring contains the reason. The implicit SIF Error values are: • Category 1 (XML Validation) • Code 2 (Message not well formed)	No
MustUnderstand	An immediate child of the SOAP Header with a "must understand attribute" was not understood. SIFHeader has such an attribute and most likely one part of this was not understood	Faultstring contains the reason. The implicit SIF Error values are: • Category 1 (XML Validation) • Code 3 (Generic Validation)	No
Server	Message could not be processed for reasons other than Message Contents	Faultstring contains the reason. The implicit SIF Error values are: • Category 10 (Transport Error) • Code 4 (Unable to establish Communication) (*) Whether to resend the original message should be determined by how the equivalent SIF_Error is handled on the HTTP/S transport.	Maybe(*)
Client	Message did not contain appropriate information.	This class of SOAP fault carries SIF Error information (Category, Code, Description and Extended Description) in the SOAP Body.	No

4.5.2 Mapping SIF elements to the SOAP Fault Header

The following table defines how to set up the element values in a SOAP Fault message being sent from an agent, when it rejects a message back to the ZIS which forwarded it originally.

SOAP Fault (Agent to ZIS)	Value of the SOAP Fault element in the body, if it exists
WS-Addressing Elements	
wsa:To	The URL of the ZIS
wsa:From	The URL of the Agent
wsa:MessageId	A unique message ID for this Fault
wsa:Action	The fixed string for all SOAP 1.1 Fault messages
wsa:RelatesTo	The Message ID of the message being rejected
<SIFHeader Elements>	
Timestamp	The time of Error message creation
SourceId	The Source ID of the Agent
OriginalSourceId	The Source ID of the Agent which originally posted the message being rejected, and **not** the ZIS. It corresponds to the SIF_OriginalSourceId element

	in SIF_Ack.

4.5.3 Mapping Web Service Transport and Messaging errors to SIF Error Category and Code

The mapping of web service related errors to existing SIF error category / error code values should be done whenever possible, because returning a category / code unique to web services "breaks" the encapsulation of the actual infrastructure supported by the receiving application's partner.

For example if the ZIS routes a Request message to an agent / application and the application is not an Object Provider, the agent should return an error category of "Request and Response" and a code of "No Provider" independent of what infrastructure it supports, rather than return an infrastructure-specific error category of "Service" and a code of "Operation Not Supported".

Where such mapping is not possible, one of the error codes in the "Generic Message Handling" category MUST be returned.

4.6 Message-specific Mapping Issues

There are several "special cases" involved in mapping the HTTP/HTTPS message schemas to their SOAP transport equivalents that are examined in greater depth here.

4.6.1 Initialization of a SIF Web Application

Register is the first message issued by the SOAP agent, and the first indication to the ZIS that the agent exists, and will be using the SOAP transport. There are two specific requirements placed on the agents issuing this message.

- The Register message MAY be directed to the HTTPS port (443), the SOAP port (880) or any other port supported by the ZIS.
- The underlying transport for this initial message MUST be SOAP/HTTPS, the secure equivalent version of HTTP v1.1

The value of the Version element in the Register message must reflect the value of the Data Model version supported by the SIF agent / application.

The individual Protocol subelements within the Register message are REQUIRED to be in accordance with the following rules.

Protocol Component	Char	Type	Meaning
Type	M	Attribute	The range of legal enumerated values in DefinedProtocolsType have been extended to include "HTTP-SOAP1.1" and "HTTPS-SOAP1.1". One of these two settings MUST be contained in the Type attribute.

Secure	M	Attribute	Unchanged in meaning Set to "Yes" (HTTPS) or "No" (HTTP) depending upon whether the underlying protocol provides a secure channel.
URL	C	Element	For all Push Mode Agents, this is the HTTP/S URL subsequently used by the ZIS to invoke operations on the "FlowControl" Web Service provided by that agent. See Section 4 for further details. It is not present if the agent registering is Pull Mode.
Property	M	Container	Consists of one or more Property Name/Value pairs, as defined below.

There are two optional Property pairs (name, value) in the Protocol element. They correspond to additional services optionally supported by the Push Mode Agent, in addition to FlowControl.

Name	Char	Value for this release
http://www.sifassociation.org/contract/DataModel/2.x	O	URL where the ZIS can access the Push Agent supported DataModel operations
http://www.sifassociation.org/contract/ZoneService/2.x	O	URL where the ZIS can access the Push Agent supported ZoneService operations

In all cases the AgentACLResponse message is returned on successful registration.

4.6.2 SOAP Pull Mode Agent issues a GetMessage

If a GetMessage operation is successfully received by the ZIS, this results in a complete Message being "packaged" and sent over the same transport the request arrived over, so that the receiving PullMode Agent can understand it. On the HTTP/S transport, the entire returned message is contained within a SIF_Message element, which in turn is contained within a SIF_Status response to the SIF_GetMessage.

On the SOAP transport, the returned message is contained in a SOAP Envelope (with both a SOAP Header and SOAP Body) which is contained in the SOAP Body of the response to the GetMessage.

The ZIS must supply whichever form the PullMode Agent can utilize, and in cases where the message was being sent over one transport and delivered over another, the entire internal message must be converted before being routed.

The following table contrasts the top level element mapping between one of the SIF Response message packets being returned to a Pull Mode Agent, sent across each of the supported transports. Note that in the case of the SOAP transport, this is a QueryResults message, which is a renamed ObjectData element.

HTTPS Transport	SOAP Transport
SIF_Message SIF_Ack SIF_Header SIF_Status SIF_Code & Description SIF_Data	SOAP Envelope SOAP Header WS-Addressing elements SIFHeader (for Status) Return Code and Description

SIF_Message SIF_Response SIF_Header Packet control elements SIF_ObjectData	SOAP Body SOAP Envelope SOAP Header WS-Addressing elements SIFHeader (for QueryResults) Packet control elements SOAP Body QueryResults

Here the SOAP Body for the Status message contains both the SOAP Header and SOAP Body of the QueryResults message it conveys (i.e. the equivalent of a SIF_Message). It is not necessary to bring the Packet Control elements of the nested QueryResults up to the actual SOAP Header for this message because:

- The individual QueryResults packets will have different Message Ids. This meets the SOAP requirement for unique Message Ids
- It matches the HTTP/S mapping, where the packetization information was in the SIF_Header of the internal SIF_Message and not the SIF_Header of the top level SIF_Status. The actual exchange sequence in this case exactly parallels the existing one.

Assuming there are no errors the exchange sequence is:

- Pull Agent issues a GetMessage to the ZIS. <In>
- ZIS synchronously responds (on the same HTTP/S connection) with a Status message to the agent, containing the SOAP envelope for the next message in the agent's Queue, in the SOAP Body of the Status. <Out>
- Pull Agent issues an "Immediate" Status message to the ZIS confirming successful reception of the new SOAP message (on a new HTTP connection). <In> The ZIS can now free up the message in the agent's Queue.
- There is no "out" part for this message, which matches the HTTP/S transport choreography.

If SMB had been requested by the Pull Agent, the exchange sequence would include "Intermediate" and "Final" Status messages in accordance with the existing HTTP/S choreography for SMB support.

4.6.3 User-level Error Occurs

There are several asynchronous "request / response" message exchange sequences supported over the SOAP transport:

- Query / QueryResponse
- ExtendedQuery / ExtendedQueryResponse
- ServiceInput / ServiceOutput

The message exchange orchestration is identical. The table below illustrates it for a Query / Query Response:

Initiating Agent	ZIS	Object Provider

Invoke Query →1	1 → Receive Query	
Receive Status ←2	2 ←Send Success Status to Agent	
	Forward Query to Object Provider →3	3 →Receive Query
	Receive Status ←4	4 ← Send Success Status to ZIS
		<Process Query and create response>
	Receive QueryResponse ←5	5←Send QueryResponse back to ZIS
	Send Success Status to Provider →6	6→Receive Status
Receive QueryResponse←7	7←Forward QueryResponse to Originating Agent	
Send Success Status →8	8→Receive Status	

In all cases, the Status message is sent back synchronously (on the same HTTP/HTTPS connection as the original message arrived on). If a transport error occurred, a SOAP Fault (with the Error Category, Error Code, and Error String contained in the SOAP Fault) is returned instead.

However there is the additional case where a higher level error occurs, causing the object provider to reject the QueryRequest, and issue an asynchronous DataModelError message instead of a QueryResponse. The DataModelError message is then asynchronously returned exactly as the QueryResponse would have been, and the SIFHeader in both cases would be identical.

Such asynchronous user errors are the only errors reported using the equivalent of the SIF_Error schema, and the Error operation must therefore be supported by both the PushAgent and ZIS web services.

Note that unlike with a QueryResponse, the SOAP Body of a Error message is determined solely by the Message namespace, and is independent of the Data Model. An example of such a Error payload is shown below:

```
<soap:Body>
 <DataModelError xmlns="http://www.sifassociation.org/messages/2.x">
      <Category>8</Category>
      <Code>1</Code>
       <Desc>No support for requested object type</Desc>
       <ExtendedDesc>A directed query has been erroneously delivered to a non-provider</ExtendedDesc>
 </DataModelError>
</soap:Body>
```

4.6.4 Packetization over the SOAP transport

The SOAP transport does not directly support message packetization. The SIF HTTP/S transport does, for the following messages types:

- QueryResponse and ExtendedQueryResponse
- SIF_ServiceOutput
- SIF_ServiceInput and SIF_ServiceNotify

Each packet includes a "PacketData" element in the SIFHeader which contains the Packet Number relative to the message and a flag indicating whether or not this is the last packet of the message.

In addition, the equivalent of a "Packet Stream Id" is required to allow the recipient to identify which message this incoming packet belongs to. The table below describes how that value is represented in each of the message types which support it.

HTTP/S Message Type that can be packetized	How a common "Stream" Id for all packets in the message is generated	SIF SOAP equivalent
SIF_Response (QueryResponse, ExtendedQueryResponse)	Common SIF_RequestMsgId set to the MsgId of the SIF_Request being responded to.	wsa:RelatesTo = MsgId of the Query / Extended Query Message being responded to. This is true for every packet in a multi-packet Response.
SIF_ServiceOutput (ServiceOutput)	Common SIF_ServiceMsgId uses the SIF_ServiceMsgId of the SIF_ServiceInput packet stream being responded to.	wsa:RelatesTo = ServiceMsgId of the ServiceInput packet stream being responded to. This is true for every packet in a multi-packet ServiceOutput.
SIF_ServiceInput (ServiceInput)	SIF_ServiceMsgId is a uniquely generated UUID for the packet stream. It is separate from each of the individual MsgIds of each packet.	There is no wsa:RelatesTo in the SOAP Header The ServiceMsgId is the link between all packets of the ServiceInput
SIF_ServiceNotify (ServiceNotify)	SIF_ServiceMsgId is a uniquely generated UUID for the packet stream. It is separate from each of the individual MsgIds of each packet.	There is no wsa:RelatesTo in the SOAP Header. The ServiceMsgId is the link between all packets of the ServiceNotify

4.6.5 Support for Bundled Event Messages

The SIF v2.6 release infrastructure includes the SIF_Events message which provides significant scalability enhancements for the SIF Zone (see the SIF Global Infrastructure Implementation Specification associated with this release).

Every Events message contains one or more internal SIF Event messages. This means the SOAP Body of an Events message contains one or more SOAP Envelopes representing the original set of Event messages which were bundled together to form an Events message.

The following table contrasts the top level element mapping between the structures of the Events message on the two SIF transports.

HTTPS Transport	SOAP Transport
SIF_Message SIF_Events SIF_Header SIF_EventMessages SIF_Message SIF_Event (Event A) SIF_Header SIF_ObjectData	SOAP Envelope SOAP Header WS-Addressing elements SIFHeader (for Events) SOAP Body EventMessages SOAP Envelope (Event A) SOAP Header

SIF_EventObject SIF_Message SIF_Event (Event B) SIF_Header SIF_ObjectData SIF_EventObject SIF_Message SIF_Event (Event C) SIF_Header SIF_ObjectData	WS-Addressing elements SIFHeader SOAP Body EventObject SOAP Envelope (Event B) SOAP Header WS-Addressing elements SIFHeader SOAP Body EventObject SOAP Envelope (Event C) SOAP Header WS-Addressing elements SIFHeader SOAP Body EventObject

The bundled Event messages can represent different types of Events (Add, Create) on different object types (StudentPersonal, EnergyUsage). As described in the Infrastructure volume, all bundled Event messages are completely independent and each may have unique values in the Header including unique Message Ids (required), but all are responded to with a single ACK.

4.6.6 Resolving the LogEntry Problem

As part of publishing any HTTP/S Event, the ZIS may have to convert the message wrapper to SOAP format if that is what is understood by a particular recipient. In the case of a LogEntry Event however the internal contents are often erroneous (since they include a copy of what caused the recipient to generate an error).

It is then an open ended problem as to how erroneous SIF_Header elements reported as part of the *Original Message* should be converted into erroneous WS-Addressing elements in the internal SOAP Header of the equivalent LogEntry Event and visa-versa. Consider the following exchanges between HTTP/S-based Application H and SOAP-based Application S that result if this were done:

- Application A publishes an Event message for an object of type X
- The ZIS translates the Event into the SOAP format and forwards it to X subscriber Application S
- Application S sees some problem with the translated SOAP header (which may originally be from Application A, or may be from the ZIS translation, or possibly subscriber S is flagging correct values as an error).
- Application S posts a LogEntry event with the original SOAP header info it believes is erroneous, contained in the SOAP Body
- The ZIS re-translates this claimed erroneous SOAP equivalent of a possibly erroneous HTTP/S header back into HTTP/S and sends it to LogEntry subscriber Application A
- Application A is left to try and figure out exactly what happened

This mapping problem would not arise if LogEntry contained an exact image of the data given to the recipient that caused the generation of the LogEntry, rather than a mandated restructuring of the original error. While

current plans call for this correction to take place in the next major SIF release, it needed to be resolved for the SOAP mapping now.

As a result, for this release, a new SoapLogEntry object type has been added specifically for the SOAP transport, that parallels the SIF_LogEntry object type on the HTTP/S transport. Unlike all other objects, the ZIS will perform NO translation of the internal contents of the original error message in a SoapLogEntry Event when mapping to the HTTP/S transport required by a subscriber, and no translation is required when going from a SIF_LogEntry to a SOAP subscriber. This serves two purposes:

- It allows a Zone Administrator looking at the unchanged SIF_LogEntry or SoapLogEntry event contents to more easily figure out exactly out what went wrong because original problem is reported in exactly the form the data had when the error was detected.

- In the case of automated processing, it avoids the open ended problems associated with analyzing erroneous data that has already been doubly translated across infrastructure boundaries.

In order not to break backward compatibility with the existing SIF_LogEntry schema on the HTTP/S platform, a separate SoapLogEntry object type was created for the SOAP platform. The two object types are mutually exclusive, and have a separate set of publishers and subscribers.

To ensure all LogEntry error events for a given Zone are collected, either an application must register and subscribe to both SIF_LogEntry and SoapLogEntry Events (one on each transport), or separate applications must share a common file or other data store.

The internal contents of a SOAPLogEntry consist of:

- The Category (1-4)
- The Code (string)
- The Application Code (string)
- The Description (string) and the Extended Description (string) of the reported error
- The OriginalMessageData where the complete original message was stored (a SOAP Envelope)

The SOAPLogEntry object XML format is provided in the *Messaging* Name Schema XSD in the SIF Web Server Developer Kit. It is replicated below.

```
<!--
        SIF_LogEntry had to be redefined uniquely here in order to avoid translating internal
        erroneous header data between transport types
-->

<xs:complexType name="SoapLogEntryType">
        <xs:sequence>
                <xs:element name="Category" minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:token">
                                        <xs:enumeration value="1"/>
                                        <xs:enumeration value="2"/>
                                        <xs:enumeration value="3"/>
                                        <xs:enumeration value="4"/>
```

```
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>

                <xs:element name="Code" minOccurs="0" type="xs:string"/>

                <xs:element name="ApplicationCode" minOccurs="0">
                    <xs:simpleType>
                        <xs:restriction base="xs:normalizedString">
                            <xs:maxLength value="64"
                                xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>

                <xs:element name="Desc" minOccurs="0">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:maxLength value="1024"
                                xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>

                <xs:element name="ExtendedDesc" minOccurs="0" type="xs:string"/>

                <!-- The next element is the original Message in its entirety.  -->

                <xs:element name="OriginalMessageData" minOccurs="0">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="OriginalMessage"
                                type="soap-env:Envelope" minOccurs="1" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="LogLevel" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="Info"/>
                        <xs:enumeration value="Warning"/>
                        <xs:enumeration value="Error"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>

    </xs:complexType>

    <xs:element name="SoapLogEntry" type="sif:SoapLogEntryType"/>
```

5 ZIS/SAWS Functionality Mapping to WSDL Interface

A web application relies on the relevant Web Service Description Language (WSDL) documents to define the interfaces of the services it may access as a web client, and the operations it must implement as a web service.

The SIF standard defines the complete set of messages the ZIS will accept and send to connected agents (and their applications), along with the order in which these messages must be invoked (choreography). For example, a SIF agent/application must first Register itself in the Zone, before it can Provision itself as a subscriber to Events subsequently posted for one or more object types (via the Event operation). Only after both operations compete successfully will the application actually be sent Event messages. This exact message set and choreography is enforced over the SOAP Transport.

There are three SIF components involved in this SOAP-based choreography:

SOAP Pull Mode Agents
Pull Mode SIF Application Web Clients (Pull SAWCs) always initiate message exchanges with the ZIS. They receive asynchronous responses to their requests by invoking the "GetMessage" operation on the appropriate SIF Infrastructure Web Services (SIWS). Since they in effect act as pure web clients (they always initiate operations and never accept incoming asynchronous HTTP/S connections), they require no corresponding WSDL.

ZIS
There are 4 SIF Infrastructure Web Services (SIWS) provided by the ZIS for accessing its messaging functionality. The set of operations provided by these 4 service interfaces (Port Types) are invoked over the SOAP transport by SIF Applications Web Clients (Pull or Push Mode Agents).

The ZIS MUST support every defined WSDL operation for all 4 SIWS interfaces and the end point URLs of all SIWS within a given Zone MUST be identical.

Push Mode Agents
There are 3 SIF Application Web Services (SAWS) that can be provided by a SOAP Push Mode Agent. One of these is required (Flow Control) and support for the other 2 (Data Service and Zone Service operations) are optional.

Push Mode Agents MUST provide responses to all invocations on every operation defined in the set of WSDLs they do support, even if the operation is illegal (ex: an incoming Query request reaching a non-object provider). In all such cases, the Push Mode Agent MUST respond by sending a SOAP message with either:

- A SIF DataModelError message with the appropriate error category and error code.
- A SOAP Fault with a faultstring value corresponding to a WS-Addressing Error "ActionNotSupported".

Every SOAP Push Mode Agent MUST provide a single WSDL end point URL within the initial Register message, where its system control Service is located. This URL is used to inform the ZIS of where to initiate an HTTP/S connection when a system control operation (ex: Sleep, Ping) is to be invoked. For these messages, the SOAP Push Mode Agent essentially acts as the Web Service, and the ZIS plays the role of the Web Client.

Any other SAWS supported by the SOAP Push Mode Agent are also specified in the Register message, as a separate Parameter element (the name of the service and its URL location) in Register/Protocol. It is RECOMMENDED that all URLs for the services provided by a single SOAP Push Mode Agent be identical.

5.1 WSDL Overview

The following principles are reflected in the design of the WSDL files for both the ZIS and the SOAP Push Mode Agents.

5.1.1 WSDL File Structure and Granularity

A given WSDL v1.1 file defines the complete external interface to a Web Service. There are 5 such WSDL files in this release.

All 5 WSDL files import a "common" file which defines the full set of WSDL messages both the ZIS and all Push Agents can exchange (Common.wsdl). Each of these messages roughly corresponds to a supported SIF_Message type in the HTTP/S infrastructure

Each WSDL file contains a single Port Type (interface), which defines a group of related operations in which these messages are exchanged. The supplied WSDLs also "bind" each of these operations to the SOAP 1.1 transport over HTTP/S.

All WSDL files have a single WSDL Service entry with an address set to the WSDL Anonymous URI. An agent developer or Zone integrator need only replace this with the URL of the actual ZIS or Push Agent-provided service used within the specific SIF Zone, before supplying it to the client as the complete WSDL description of the service it will be accessing.

Additionally, any Push Mode Agent providing a service MUST supply the endpoint URL of that service at Registration time.

5.2 WSDL Versioning and Namespaces

The version of both the ZIS and Push Agent WSDL files is defined as follows:

The specific major and minor release numbers of the SIF messaging schema MUST be contained in the "documentation" subelement within "definitions".

```
<documentation>Version 2.6</documentation>
```

The "compatibility" version number should also be appended to the back of the WSDL target namespace URL.

For the WSDL files that define both the ZIS and Push Mode Agent Web Services, the following Target Namespaces were used:

WSDL Functionality	Namespace
Administration & Provisioning (ZIS only)	http://www.sifassociation.org/contract/Administrate_Provision/2.x
Message Flow Control (Push Agent only)	http://www.sifassociation.org/contract/FlowControl/2.x
Object Data Exchange (ZIS and Push Agent)	http://www.sifassociation.org/contract/DataModel/2.x

Zone Service Support (ZIS and Push Agent)	http://www.sifassociation.org/contract/ZoneService/2.x
Pull Mode Agent Queue Management (ZIS only)	http://www.sifassociation.org/contract/QueueManagement/2.x

With respect to versioning, unlike "standard" web services which are not backward compatible if extended (particularly where there are additional elements in the newer response); SIF web service clients must be equivalent to SIF v2.x object provider clients.

This means they MUST ignore any elements in an incoming XML messages that are not defined in their schema set (i.e. where the XML will not validate) when the schema version is later than the one the web service was written to accept.

5.3 WSDL Operations

There is roughly a 1:1 correlation between the types of SIF messages defined for the HTTP/S transport, and the WSDL operations defined within the Port Types of all ZIS and Push Agent web services. The WSDL operation name is located in the SOAP Header in element wsa:Action. All the data corresponding to that operation is located under the single child element of the SOAP Body.

5.3.1 Legal WSDL Operations

The following table defines the legal WSDL operations that comprise the Port Type interfaces of both the SIF Infrastructure Web Services (SWIS) provided by the ZIS, and the SIF Application Web Services (SWAS) provided by the SOAP Push Mode Agent. In several cases a single HTTP/S SIF_Message Type was broken open and its subtypes mapped to separate WSDL operations. For example, a SIF_ACK is represented on the SOAP transport by either a Status or Error message.

All WSDL operations are defined for the SIWS and MUST be supported by the ZIS. WSDL Operations with "(*)" are defined for the SAWS as well.

SIF Message Component	WSDL Operation	Comment
SIF_Ack/SIF_Status	Status	Issued to the SIWS (ZIS) only. Also serves as the synchronous output method for every other operation (see Note below)
SIF_Ack/SIF_Error SIF_Response/SIF_Error SIF_ServiceNotify/SIF_Error	DataModelError(*)	SIF_Error subelements are retained (without the "SIF_" prefix). In case of transport errors, the elements are mapped to SOAP Fault.
SIF_Event/SIF_ObjectData/SIF_EventObject	Event(*)	Object Name and Object Action moved to SIFHeader
SIF_BundledEvents	Events	Introduced in SIF v2.6, this is a set of bundled Event messages, packaged together in one "Events" message. Over the SOAP transport this results in multiple SOAP Envelopes within the SOAP Body of this message, under the top level element "EventMessages".

SIF_Provide	Provide	
SIF_Provision	Provision	
SIF_Register	Register	
SIF_Request/SIF_Query	Query(*)	Was Request subtype now separate operation
SIF_Request/SIF_ExtendedQuery	ExtendedQuery(*)	Was Request subtype now separate operation
SIF_Response/ObjectData	QueryResults(*)	Was Response subtype now separate operation **Note: Name changed for clarity**
SIF_Response/ExtendedQueryResults	ExtendedQueryResults(*)	Was Response subtype now separate operation
SIF_Subscribe	Subscribe	
SIF_SystemControl/SIF_SystemControlData		
/SIF_Ping	Ping (*)	Empty contents
/SIF_Sleep	Sleep (*)	Empty contents
/SIF_Wakeup	Wakeup (*)	Empty contents
/SIF_GetMessage	GetMessage	Issued by Pull Mode Agent (pure web client) only. Synchronous Status response (on same HTTP/S connection) MUST contain the complete SOAP Envelope of next message within its SOAP Body.
/SIF_GetZoneStatus	GetZoneStatus	ZoneStatus object data is returned in the synchronous ZoneStatusResponse message.
/SIF_GetAgentACL	GetAgentACL	AgentACL object data is returned in the synchronous AgentACLResponse message.
/SIF_CancelRequests	CancelRequests(*)	Internal list of Message IDs is not replicated in SOAP Header.
/SIF_CancelServiceInputs	CancelServiceInputs(*)	Internal list of ServiceMessage IDs is not replicated in SOAP Header.
Synchronous Response Messages		
	SOAP_Envelope	Synchronous successful response to GetMessage
	ZoneStatusResponse	Synchronous successful response to GetZoneStatus
	AgentACLResponse	Synchronous successful response to GetAgentACL
SIF_Unprovide	Unprovide	
SIF_UnRegister	Unregister	Empty contents.
SIF_Unsubscribe	Unsubscribe	
SIF_ServiceInput/SIF_Body	ServiceInput (*)	Contents of ServiceInput element in SOAP Body are xs:Any
SIF_ServiceInput/SIF_Error	SOAP Fault	
SIF_ServiceOutput/SIF_Body	ServiceOutput (*)	Contents of ServiceOutput element in SOAP Body are xs:Any
SIF_ServiceOutput/SIF_Error	SOAP Fault	
SIF_ServiceNotify/SIF_Body	ServiceNotify (*)	Contents of ServiceNotify element in SOAP Body are xs:Any

.

5.4 SIF Operation to WSDL Service Mapping.

The set of SIF operations are divided into the following Web Services. Each web service is defined with a single WSDL file, containing a single Port Interface and Port Binding.

Web Service Name	Operations	Support
Administrate_Provision	*Register, Unregister, Provide, Unprovide, Subscribe, Unsubscribe, Provision, GetZoneStatus, GetAgentACL, Ping, Sleep, Wakeup*	*ZIS only*
FlowControl	*Ping, Sleep, Wakeup*	*Push Mode Agent only*
DataModel	*Event, Events, Query, ExtendedQuery, QueryResults, ExtendedQueryResults, CancelRequests, DataModelError*	*ZIS and optionally Push Mode Agent*
ZoneService	*ServiceNotify, ServiceInput, ServiceOutput, CancelServiceInputs, ZoneServiceError*	*ZIS and optionally Push Mode Agent*
QueueManagement	*GetMessage, GetMessageStatus*	*ZIS only*

Each of these web services are further defined in the sections below. In all cases, the web client invoking any of the operations in a web service provided by a SOAP Push Mode Agent must be the ZIS, and any web client invoking any of the operations in a web service provided by the ZIS must be a SOAP agent (Pull or Push Mode).

5.4.1 Administrate_Provision (ZIS support only)

This SIF Infrastructure Web Service (SIWS) encapsulates the set of ZIS operations which can be invoked by SIF Application Web Clients (SAWCs) to:

- Register and provision themselves as users and suppliers of Zone resources
- Access Zone administrative data
- Control the flow of messages from all ZIS Services.

All the operations MUST be supported by the ZIS.

The meaning of each operation and the required choreography between them exactly parallel the equivalent ZIS-supported methods provided over the HTTP/S transport. The differences are:

- The PushMode SOAP Agent MUST provide the complete list of Web Services it supports and their respective end point URLs, within the Protocol element of the initial Register message. It is RECOMMENDED that the values of all such end point URLs be identical.

- AgentACLResponse and ZoneStatusResponse messages are defined in the WSDL as the “out” part of the GetAgentACL and GetZoneStatus operations, and in a manner identical to the HTTP/S transport, are returned synchronously on the same HTTP/S connection the corresponding GET operation was invoked on. The Messaging schema defines these messages as type “xs:any” so that like all other objects, the schemas for AgentACL and ZoneStatus must be taken from the appropriate Data Model namespace.

- An entire SOAP Envelope (SOAP Header and SOAP Body) is defined as the "out" part of the GetMessage WSDL operation, and is returned synchronously on the same HTTP/S connection the GET operation was invoked on. This parallels how a SIF_Message is returned over the HTTP/S transport.

5.4.2 FlowControl (Push Agent support only)

The three operations in this Web Service provide the ZIS client with the network level functionality needed to control the flow of all incoming asynchronous messages from the PushMode Agent. Its support is mandatory.

The ZIS can assert these controls for a Pull Mode Agent as well, via the message it returns in response to a "GetMessage" operation.

5.4.3 DataModel (Both ZIS and (optionally) Push Mode Agent support)

The operations contained in the DataModel web service are supported by the ZIS (mandatory) and by SOAP Push Mode Agents (optional). They are invoked on the ZIS by SOAP agents to post Events, make Query and Extended Query requests, and (in the case of Object Providers) to supply Query and Extended Query responses.

They are invoked by the ZIS on SOAP PushMode Agents which support this web service, as the final step in its delivery of these messages to the correct destination. The ZIS can deliver these messages to Pull Mode Agents as well, via the message it returns in response to a "GetMessage" operation.

They encapsulate the full (and identical) set of CRUD operations used by agents and the ZIS to support access to all SIF Object Data.

5.4.4 ZoneService (Both ZIS and (optionally) Push Mode Agent support)

The operations contained in the ZoneService web service are supported by the ZIS (mandatory) and by SOAP Push Mode Agents (optional). They are invoked on the ZIS by SOAP agents to access or support one or more SIF-defined Zone Services (such as the Assessment or Student Record Exchange Zone Services defined in the US release of SIF US v2.4).

They are invoked by the ZIS on SOAP PushMode Agents which support this web service, as the final step in its delivery of Zone Service messages to the correct destination. The ZIS can deliver these messages to Pull Mode Agents as well, via the message it returns in response to a "GetMessage" operation.

The operations of this Web Service map directly to the three message types comprising the infrastructure expansion made in support of Zone Services (ServiceInput, ServiceOutput, ServiceNotify).

The supported operations in this WSDL standardize the interface to all SIF Zone Services in the same way the interface to all Object Providers conforms to the DataModel WSDL. Any SIF Zone Service (of whatever variety) is only accessible through operations in this Web Service group at the message level. Any "operation" (and associated arguments) specific to an individual Zone Service are embedded in the SOAP Body. Specifically the WS-Addressing "Action" value will be either ServiceInput, ServiceOutput or ServiceNotify.

Defining and externalizing unique WSDL definitions for each Zone Service type is a future enhancement to the SIF standard.

5.4.5 Queue Management (ZIS support only)

Only the GetMessage and Status operations are provided by this ZIS web service. It is used exclusively by Pull Mode SIF Web Agents. It allows them to function solely as web clients, by supporting an operation which "gets the next message to be delivered". This operation SHOULD be invoked synchronously by these clients only when they are ready to receive a new message, and it eliminates the need for them to provide a call back URL at Registration time, or to support any SIF-provided WSDL.

There is a message exchange choreography required between the client and ZIS which is unique to this web service.

- A SOAP Pull Mode Agent invokes the GetMessage operation of the QueueManagement Web Service
- It synchronously (on the same HTTP/S connection) receives a SOAP message in which the SOAP Body contains the entire SOAP Envelope of the returned message.
- The agent then invokes the Status operation of the QueueManagement Web Service, letting it know whether it can release the previously sent message from the delivery queue for this Pull Mode Agent.

The final part of that exchange represents the only time that a SIF Status message is issued asynchronously, rather than being a synchronous response to an operation invocation.

5.5 Registration of Multi-Service SOAP Push Mode Agents

The initial Register message from a SOAP Pull Mode Agent is quite similar to that sent by a Pull Mode Agent registering over HTTP/S. The sole difference is in the Protocol subelement, where the Type is one of either:

- HTTP-SOAP1.1
- HTTPS-SOAP1.1

A SOAP Push Mode Agent on the other hand has to inform the ZIS which SIF Application Web Services (SAWS) interfaces it is supporting, and where the end points are. There are three defined SAWS:

- FlowControl (Required)
- DataService (Optional)
- ZoneService (Optional)

The URL subelement in Register corresponds to the end point of the FlowControl web service supplied by the SOAP Push Mode Agent. Support for either or both of the other two services is optionally indicated in Name / Value pair entries of the Register/Properties list, where the Name is the Service name (postfixed with "URL"), whose interface is supported by the SOAP Push Mode Agent, and the Value is the corresponding end point URL where that Service is located.
The full set of Register Properties is shown below.

Name	Char	Value for this release

InfrastructureVersion	M	"2.6" (packaging of Transport and Message namespaces)
DataModelURL	O	The value of the Register/URL element
ZoneServiceURL	O	The value of the Register/URL element

It is RECOMMENDED that the end point URL for the DataModel and/or the ZoneService supported by the SOAP Push Mode Agent be identical to that of the FlowControl Service (i.e. that all supported agent operations invoked by the ZIS are dispatched to the same URL). All ZIS-provided Web Services have the same End Point URL.

An example of a Register message sent by a SOAP Push Mode Agent which supports the Data Service (at the same URL as the FlowControl Service) but not the ZIS Service, is shown below. Note the URLs of the two supported services are identical.

```
<soap:Body>
 <Register xmlns="http://www.sifassociation.org/messages/2.x">
      <Name> Acme Agent for WAP 2.x</Name>
      <Version>2.6</Version>
      <MaxBufferSize> 524288</SIF_MaxBufferSize>
       <Mode>Push</Mode>
       <Protocol Type="HTTPS-SOAP1.1" Secure="Yes">
                <URL> https://AcmeHost:8030/StudentAdmin</SIF_URL>
                <Property>
                      <Name> InfrastructureVersion</Name>
                      <Value> 2.6</Value>
                     <Name> http://www.sifassociation.org/contract/DataModel/2.x </Name>
                      <Value> https://AcmeHost:8030/StudentAdmin</Value>
                 </Property>
       </Protocol>
       <NodeVendor>AcmeGadgetsInc</NodeVendor>
       <NodeVersion> >2.0.1.20</SIF_NodeVersion>
       <Application>
              <Vendor>CoyoteEnterprises</Vendor>
              <Product> Web Administration Portal 5.x</SIF_Product>
              <Version>5.1.2</SIF_Version>
       </Application>
 </Register>
</soap:Body>
```

6 HTTP/S Interoperability

The Basic Profile 1.2 mandates a set of requirements on HTTP/S running under SOAP, some of which vary from the way SIF uses HTTP/S directly as the transport protocol. These differences are summarized in the following table and wherever possible SHOULD be visible only to the ZIS, which must bridge the following differences.

In most cases the necessary conversions are straightforward. Where they are not, they are expanded.

Situation/Protocol Aspect	Existing SIF HTTP/S Transport	BP1.2 - HTTP/S under SOAP
Successful Reception of Message	Only a 200 “OK” HTTP status code	Either a 200 “OK” or 202 “Accepted” HTTP status code
HTTP Errors (non-SOAP)	The SIF HTTPS protocol uses the 200-OK response notice to communicate all responses If a client receives any 3xx, 4xx, or 5xx response notices, it should treat these responses as if a transport error has occurred.	"400 Bad Request" HTTP status code, if a HTTP Request message is malformed "405 Method not Allowed" HTTP status code if a HTTP Request message's method is not "POST" 415 Unsupported Media Type" HTTP status code if a HTTP Request message's Content-Type header field-value is not permitted by its WSDL description.
SOAP Errors	The SIF HTTPS protocol uses the 200-OK response notice to communicate all responses If a client receives any 3xx, 4xx, or 5xx response notices, it should treat these responses as if a transport error has occurred.	500 "Internal Server Error" code only SOAP Fault contained in SOAP Body, **<This requires a conversion between HTTP/S codes 200 and 500 when a SIF_Error is being converted to a SOAP Fault message.>**
HTTP Connection Strategies <Note: Since all applications are connected through the ZIS, the ZIS may support differing connection strategies depending upon the transport protocol used to communicate with the partner.>	Use persistent connections. The client may send additional POST requests and receive the HTTP responses using the same connection. Clients SHOULD use persistent connections for performance reasons but MUST be able to use non-persistent connections if the server does not wish to use persistent connections	An HTTP/S connection is guaranteed to persist across only a single WSDL operation (input and output message exchange). This exchange is thus “synchronous. Only the anonymous URI may be used as the address for wsa:ReplyTo and wsa:FaultTo to ensure a synchronous response is sent back to the wsa:From endpoint on the existing HTTP/S connection.
Character Encoding	UTF-8	UTF-8
Message Compression/Encoding	It is RECOMMENDED that servers return a 406 (Not Acceptable) status	The set of content-encodings allowed by HTTP is open-ended and any

	when a requested encoding cannot be negotiated. If a 406 is received, the client SHOULD assume compression using the specified algorithm(s) is not supported and retry communication as per SIF HTTPS Transport or SIF HTTP Transport It is RECOMMENDED that servers unable to process a particular content encoding return a 415 (Unsupported Media Type) status code.	besides 'gzip', 'compress', or 'deflate' are an extensibility point. There are no fixed error codes reserved for compression/encoding errors. MTOM is not supported in this release.
HTTP soapAction Attribute	N/A	When present, this value MUST agree with the value of the wsa:Action element in the SOAP Header.