CEN

WORKSHOP

AGREEMENT

CWA 14928

March 2004

ICS 35.080; 35.240.99

English version

Review on SIF Infrastructure, Architecture, Message Processing and Transport Layer

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: rue de Stassart, 36 B-1050 Brussels

Ref. No.:CWA 14928:2004 E

Contents

Foreword

This work is directly related to the European Commission's Mandate M/280 "Standardisation mandate to CEN, CENELEC and ETSI in the domain of 'Learning and Training Technologies & Educational Multimedia Software', covering the development of a workplan for standards related activities in relation to Learning Technologies. The first agreed workplan was published as CWA 14040 in the year 2000. In a second step, the original workplan has been expanded. The development of Interoperability frameworks for exchange of information between diverse management systems in cooperation with OASIS [1] was one of the recommendations for further work.

This CWA was prepared by an appointed Project Team within the CEN/ISSS Learning Technologies Workshop. The CEN/ISSS Learning Technologies Workshop (WS-LT) [2] agreed at its meeting in Madrid on April 04/05 2002 to provide interoperability specifications for a range of information exchange, initially aimed at school systems, but also applicable in a wider lifelong learning context. This work item will support the output of the OASIS (Open Architecture and Schools in Society) project which enables different applications and systems to share information.

OASIS is an ambitious project to promote virtual communities in the school system, partly by harmonizing information exchange standards. It has been proposed in the framework of the User-friendly information society (the IST Programme) under the Fifth Framework Programme, which defines the European Commission activities in the field of RTD (Research Technological development and Demonstration). The work of OASIS will initially be based upon the SIF project (Schools Interoperability Framework) [3] but will be adapted by the partners to meet European needs. Some of the original specifications will be discarded and some additional ones will be created.

This CEN Workshop Agreement makes a revision of the main parts of the SIF specification [4] including Message Passing, Architecture, Infrastructure, and the transport layer. Based on this study it proposes a set of modifications on the original SIF specifications to enable the development of interoperability frameworks for exchange of information between diverse management systems within an European context. This was possible thanks to a close collaboration between the OASIS and the CEN/ISSS WS-LT.

The document has been developed through the collaboration of a number of contributing partners, representing a wide mix of interests, from universities to commercial companies representatives. The names of the individuals and their affiliations that have expressed support for this CWA is available form the CEN/ISSS Secretariat

The final review/endorsement round for this CWA was started on the 2004-01-12 and closed 2004-01-25.

The final text of this CWA was submitted to CEN for approval and publication in 2004-02-04.

1 Scope

The purpose of this CWA is to review several parts of the SIF specification: Message Passing, Architecture, Infrastructure and the transport layer. Based on this analysis a set of recommendations are proposed to enable de development of interoperability frameworks for exchange of information between diverse management systems.

Although most of the outcomes from this CWA are applicable in a worldwide context, special emphasis was put to meet the particular needs of a European context and assure that the latest technical standards are taken into account.

This work will support the outputs of the OASIS (Open Architecture and Schools in Society) project that enable different applications and systems to share information. This CWA is intended to be used by OASIS and other initiatives worldwide which would like to localise the SIF specification in their particular contexts, and to develop their own interoperability frameworks, taking into account standards/specifications that may be reused and the latest technologies available.

Other people and organizations that may be affected by the recommendations made in this report are schools, teachers, software enterprises developing SIF-compliant software and software managers in schools.

2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

API	*Application Programming Interface*
BPEL4WS	*Business Process Execution Language for Web Services*
CEN	*European Committee for Standardization (Comitée Européen de Normalisation)*
CENELEC	*European Committee for Electrotechnical Standardization*
CWA	*CEN Workshop Agreement*
ETB	*European Treasury Browser*
ETSI	*European Telecommunications Standards Institute*
EUN	*European SchoolNet*
IDL	*Interface Definition Language*
IEC	*International Electrotechnical Commission*
IEEE	*Institute of Electrical and Electronics Engineers*
ISO	*International Organization for Standardization*
ISSS	*Information Society Standardization System*
JMS	*Java Message Service*
MOM	*Message Oriented Middleware*
OASIS	*Open Architecture and Schools in Society*
OCSL	*Open Code Software Library*
RFC	*Request For Comments*
RPC	*Remote Procedure Call*
SIF	*Schools Interoperability Framework*
SOAP	*Simple Object Access Protocol*
UML	*Unified Modelling Language*
W3C	*World Wide Web Consortium*
WS/LT	*CEN/ISSS Workshop on Learning Technologies*
XML	*eXtensible Markup Language*
ZIS	*Zone Integration Server*
ZMS	*Zone Management Server*

3 References

[1] OASIS, *Open Architecture and Shools in Society.* Information available in http://oasis.cnice.mecd.es

[2] CEN ISSS WS-LT, CEN *Information Society Standardization System Learning Technologies Workshop* http://cenorm.be/isss/Workshop/LT/Default.htm

[3] SIF, *Schools Interoperability Framework*. Information available in http://www.sifinfo.org

[4] SIF Specification v1.1. Information available in http://www.sifinfo.org/spec.html

[5] UML, *Unified Modeling Language.* Information available in http://www.uml.org

[6] BPEL4WS, *Business Process Execution Language for Web Services.* Information available in ftp://www6.software.ibm.com/software/developer/library/ws-bpel11.pdf

[7] SOAP, *Simple Object Access Protocol*. Information available in http://www.w3.org/TR/SOAP/

[8] IDL, *OMG's Interface Definition Language*. Information available in http://www.omg.org/cgi-bin/doc?formal/01-02-07

[9] European Schoolnet EUN http://www.eun.org

[10] ETB, *European Treasury Browser* http://www.en.eun.org/eun.org2/eun/en/etb/content.cfm?lang=en&ov=7208

4 Introduction to SIF

The Schools Interoperability Framework (SIF) [3] is an industry initiative to develop an open specification for ensuring that school learning and administrative software applications work together more effectively. SIF is not a product, but rather an industry-supported technical blueprint for school software that will enable diverse applications to interact and share data seamlessly, now and in the future.

Education is facing a critical challenge in deploying technology due to the pressing problem of interoperability. Today, applications available for schools and their districts are either closed systems or systems that allow customer access only through proprietary interfaces and data formats.

Since 1997, education software companies, school district technology coordinators, and administrators have met to discuss ways to answer the interoperability challenge. Their solution: the Schools Interoperability Framework, a specification to:

- Define standard formats for shared data (e.g., student demographics information)
- Define standard naming conventions for this shared data
- Define the rules of interaction among software applications

Open Architecture and Schools in Society (OASIS) is a project with a clear ultimate objective: to promote small virtual communities in schools within the public educational system. For this OASIS uses SIF as the basis to develop a European version of this specification.

In particular, OASIS will pay special attention to the role of socialisation in traditional schooling, as one of the main assets of the public school system, and focus on how this role can be enhanced with the help of the Internet.

This final educational outcome requires the development of a technical network. It consists of setting up both local and regional open telematics and informatics architecture for the school educational network together with zonal servers available to meet the needs of automated maintenance of these school networks.

Its aim is simultaneously to integrate a set of data models together with the organisational structures of this data, thus favouring the interoperability between educational networks.

This document includes a review of the SIF Infrastructure, Architecture, Message Processing and Transport Layer by the OASIS project and a set of recommendations based on this review. The base document for this review was the SIF specification [4], where the SIF model is presented and defined. This review was taken from four OASIS deliverables:

1. SIF Message Passing (see appendix A, section 8.1)
2. SIF Architecture (see appendix B, section 8.2)
3. SIF Infrastructure (see appendix C, section 8.3)
4. HTTPS and its use by the SIF Infrastructure transport layer (see appendix D, section 8.4)

The present document was developed to provide a further insight into the SIF Infrastructure, Architecture, Message Processing and Transport Layer. Using the above mentioned OASIS deliverables as the basis, this document has the following objective: Identify key points and recommendations for the SIF community.

5 SIF Infrastructure, Architecture, Message Processing and Transport Layer

This section includes an introduction to the SIF Infrastructure, Architecture, Message Processing and Transport Layer. It is based on the SIF specification itself [4] and the deliverables by the OASIS included as appendixes (section 8).

5.1 SIF Message Passing

SIF is implemented as a Message Oriented Middleware (MOM) that has standardised on specific types of messages and transport protocols in the Administrative space. All of the messages, transport, and Application Programming Interfaces (APIs) are defined. SIF specifies a Zone Integration Server (ZIS) that acts as the message router and queues messages for delivery. SIF provides an API for creation of Agents that can handle all communication with the ZIS. SIF is not a general purpose Messaging Service, however it can be used in conjunction with others as long as the Agents are programmed that way.

An example of a typical SIF implementation would be to connect products from different vendors together within a single school. These applications might include a student information system (SIS) application, a food service program and a library automation application. Each of these applications will need to have a vendor provided interface program, for communication with the ZIS, called an "Agent." Agents are custom applications that act as the go between for the proprietary application and the SIF framework.

The data that can be exchanged in SIF is pre-defined using XML data objects. These XML objects define the semantics of information that can be managed by the applications. As an example the following are defined in the SIF:

- StudentSchoolEnrollment,
- StudentPersonal,
- StaffPersonal
- Address

All SIF interchanges must use these pre-defined data objects and agents must exist to transform internal proprietary data formats into the pre-defined objects and interface with the ZIS for the exchange of data. Messages that do not fit the SIF standard will generate an error.

5.2 SIF Architecture

5.2.1 Basics

Although there are many variations of SIF topographies, the common feature is that a number of applications wish to share data. SIF groups applications wishing to share data together into a logical entity which is called a *Zone*. There are no predefined sizes for zones, so a zone can be as large or small as required in order to meet the needs of the customer. The size of the zone is flexible and could consist of a single building, school, a small group of schools, a district, etc.

All SIF implementations regardless of their complexity consist of one or more applications with their associated interface *agents* all being managed by a *Zone Integration Server* (ZIS). Each application must have an *agent* to communicate with the ZIS. Agents are typically provided by the application vendor. An agent never talks to another directly. All communications between agents are managed by the ZIS.

A Zone Integration Server is a program that provides integration services to all the agents registered with it so that they can provide data, subscribe to events, publish events, request data, and respond to requests. It is responsible for all access control and routing within the system.

A SIF implementation must enable different applications to exchange data efficiently, reliably, and securely regardless of what platforms are hosting the applications.

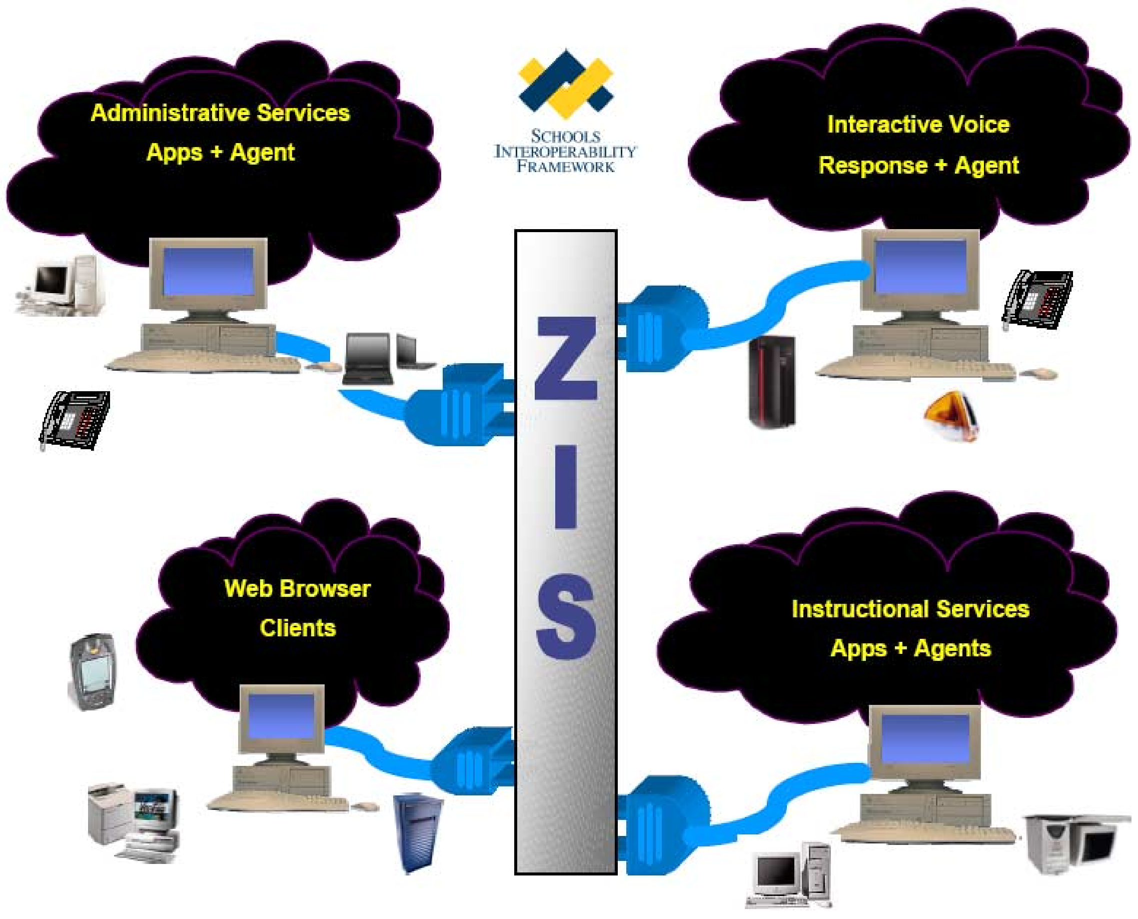


Figure 1: SIF Architecture

5.2.2 Data representation: XML

Data that can be exchanged in SIF is defined as a series of data *objects* defined using XML. These data objects are carried within a SIF Infrastructure message when they are being transferred between agents.

5.2.3 Retrieval of information: Request/Response

Applications can query information from specific SIF data objects in a Request/Response fashion. The requester's agent sends a message to the ZIS, which will forward the request to the destination agent. If the destination of the requester is not specified, the ZIS will send the request to the *provider* for the object being queried. Thus, the provider of an object is the default responder application for requests to that object.

To act as a provider, an application must notify this to the ZIS. There is a *single* provider per object per zone, but there may be *multiple* responders for a given object in a zone.

5.2.4 Data updates: Publish/Subscribe

Applications propagate data updates by publishing *events* that are reported to other agents that have shown their interest in data updates. Events are sent to the ZIS by the generating agent, and the ZIS is responsible for delivering that event to the interested parties without any further communication to the event originator. Defined events are: “Add”, “Update” and “Delete”.

In order for an application to receive an event, *subscriptions* for the SIF data objects of interest must be entered at the ZIS. In contrast no action is required to publish events. The application must only be registered at the ZIS, a required task for each application.

Many applications may publish events for a given data object

5.2.5 Communication: asynchronous model

The reliability requirement implies that when an application sends out a message to a receiver, the delivery of the message is guaranteed. This requirement also implies that messages sent by an application will arrive at the receiver in the same order as they are sent and that each message is delivered once and only once.

To ensure scalability and reliability, all communications between agents is asynchronous. An agent cannot be assured that it will get an immediate response to a message it sent. When an agent is not connected to the SIF environment, other agents can still send messages to it, as the ZIS will deliver them as soon as the agent is available. A message queue solution is suggested in the specification, and should feature a high availability and reliability implementation (e.g. permanent storage, so that messages cannot be lost).

Two models for delivering messages to an agent are defined:

- *Push* model refers to the action by a ZIS actively to deliver messages to an agent without the agent having to initiate contact with the ZIS.
- *Pull* model refers to the action by an agent explicitly to request a single message from the ZIS.

5.2.6 Security

Providers of sensitive data must never expose that data over an insecure channel so that a secure model is therefore required. The SIF security model centres around three areas: encryption, authentication and access control.

- Encryption provides the mechanism to ensure that only the sender and receiver of a message can view the message contents. All agents and ZIS implementations must support the SIF HTTPS protocol, which provides secure transport, with encryption of the data being exchanged. Other transports may be used, and, if so, it is important to know if they are secure to avoid exposing sensitive data. The ultimate responsibility for guaranteeing the security of data lies with the originating agent or zone administrators. Several levels of encryption are defined: no encryption, encryption with symmetric key length of 40, 56, 80 and 128 bits.
- Authentication. The aim of authentication is to provide a means to ensure that the author of a message is the actual author. Another requirement is message integrity. Authentication support is optional, but highly recommended. The SIF HTTPS protocol supports authentication between an agent and a ZIS. If authentication is enabled, a message receiver can trust the SIF HTTPS implementation to verify that the message in its entirety comes from the claimed sender.
- Access control. Its target is to determine which entities have the permissions required for the messages that the agent sends. The SIF specification defines an Access Control List behaviour. Access Control should deal with agent permissions associated to provision, subscription, request, response, and events. A ZIS will exhibit behaviour with regard to the ACL that could be perceived by an Agent as if a virtual table exists defining the following information:

Field	Comments
Agent name	The unique ID for an Agent (provided as the Source ID in a SIF_Register message)

Object name	The Object being manipulated (e.g., StudentPersonal, etc.)
Provide	Can this Agent register as the provider for this object?
Subscribe	Can this Agent register as a subscriber for this object?
Report “Add” event	Can this Agent report "Add" events for this object?
Report “Update” event	Can the Agent report "Update" events for this object?
Report “Delete” event	Can the Agent report "Delete" events for this object?
Request object	Can this Agent request information for this object?
Respond to request for object	Can this Agent respond to a request for this object?

Table 1: Access control in SIF

5.3 SIF Infrastructure

According to the SIF specification, Infrastructure develops and specifies a standard framework of messages and communication mechanisms that allow diverse applications in the education industry to effectively, reliably, and securely exchange information over open networks in a platform-neutral manner.

Section 4 of the SIF specification is devoted to infrastructure details. SIF infrastructure consists of:

- Elements.
- Messages.
- Objects.

5.3.1 Elements

There are two elements that are common to all SIF messages:

- SIF_Header. Common message header for all SIF messages. It carries information related to message identification, date and time associated with the message, source identification, destination identification, and security information.
- SIF_Message. Outer most tag in all SIF messages. It has an attribute specifying the XML [4] namespace for SIF messages.

5.3.2 Messages

These messages are defined in SIF:

- SIF_Ack. Acknowledgement of infrastructure messages. All infrastructure messages will return a SIF_Ack as a result to indicate if the request was successful or not. If successful, the SIF_Ack can include additional information to be sent to the caller. If an error is reported, the SIF_Ack will include one or more SIF_Error elements.
- SIF_Event. This is used to deliver event (add, change, delete) objects. It includes the actual objects (partial or whole) that are sent along with the SIF_Event.
- SIF_Provide. This is used for advertising the provision of data objects.
- SIF_Register. This is used for agent registration with a ZIS. Registering is necessary before sending out any message. SIF_Register includes identification information and all relevant information for the ZIS to contact the agent: SIF version, buffer size, mode (push/pull), protocol information, etc.
- SIF_Request. This is used to request information in SIF data objects from other agents. It specifies fields in the object to request and the query or search criteria. Support for specification of conditions to be satisfied by the requested objects is provided: complex queries are built by using AND and OR Boolean operators. Each simple condition is tested by using an operator which, in the current SIF specification (1.1) is restricted to equality (EQ).

- SIF_Response. This is used to respond to a SIF_Request message. It consists of several data objects, which are complete or partial. SIF_Error elements may be present if any error conditions have occurred while processing the SIF_Request. A response may be divided into multiple SIF_Response messages only if the response is so long that sending it in a single message may cause performance problems or longer latency.
- SIF_Subscribe. This is used to indicate subscription to a specified object.
- SIF_SystemControl. This is used to control the flow of data between two SIF nodes. It is a container for specialised control messages. These include: SIF_Ping, to detect if an agent or ZIS is ready to receive and process messages; SIF_Sleep, used to signal the receiver that it must not send any more messages to the SIF_Sleep sender; SIF_Wakeup, used to signal that a sleeping agent or ZIS is ready to resume message processing; SIF_GetMessage, to provide an agent with a mechanism for pulling a message from a ZIS.
- SIF_Unprovide. This removes the message sender as a provider of the data object contained in the message.
- SIF_Unregister. This allows an agent to remove any association with a ZIS. The ZIS will remove all provisions and subscriptions it maintains for the sender.
- SIF_Unsubscribe. This removes the message sender as a subscriber to the SIF_Events contained in the message.

5.3.3 Objects

There is only one Infrastructure object defined:

- SIF_ZoneStatus. This is an object implicitly provided by all Zone Integration Servers to provide information about the ZIS. It includes such information as ZIS vendor name, product and version, zone name and identifier, and information about the zone status:
  - Registered providers and the objects provided by each provider.
  - Registered subscribers and the event objects subscribed by each subscriber.
  - All nodes attached to the ZIS: name, identifier, type, communication mode (push/pull), communication protocol and associated security, message processing status (sleeping), maintained statistics (optional, and with no defined child elements in version 1.0).
  - ZIS supported protocols, security mechanisms, and SIF versions.

That is, this SIF infrastructure object can be envisaged as a container for zone/ZIS monitoring information.

5.4 HTTPS and its use by the SIF Infrastructure transport layer

5.4.1 Introduction

The Hypertext Transfer Protocol over Secure Socket Layer (HTTPS, or HTTP over SSL) is a web protocol originally developed by Netscape and built into web browsers that encrypts and decrypts user page requests as well as the pages that are returned by the web server. HTTPS is the use of Secure Socket Layer (SSL) for web connections via a sub-layer under a normal HTTP application layering. HTTPS uses port 443 instead of HTTP port 80 in its interactions with the lower layer at the TCP/IP level.

5.4.2 SIF Infrastructure and HTTPS

The SIF Infrastructure messages are used by SIF to encapsulate and transfer data objects. Together, they form a messaging application program interface (API), which is expressed in XML.

The SIF Infrastructure depends upon the transport layer to provide a reliable connection to move messages back and forth between client and server. The transport layer is also responsible for providing data security by means of data encryption and authentication of the client and server. Some transport layers even provide data compression, which is an important factor when processing a large volume of XML messages.

Different types of transports are or will become available providing various features and benefits. An Agent or Zone Integration Server (ZIS) may employ multiple transport protocols but they must support SIF HTTPS.

5.4.3 The SIF HTTPS Transport

In order to ensure that Agents and Zone Integration Servers can communicate with each other regardless of vendor or platform, all Agent and ZIS implementations must support the SIF HTTPS transport layer protocol.

SIF HTTPS combines the HTTP 1.1 protocol (RFC-2616) and the TLS 1.0 secure socket protocol (RFC-2246) resulting in an easy to use and secure transport protocol.

5.4.3.1 SIF HTTPS Request and Response Model

A client is defined as the party (Agent or ZIS) which initiates a connection to a remote machine. The remote end is known as the server. A client using the SIF HTTPS protocol opens a connection to the server and sends a HTTP 1.1 POST request with the SIF Infrastructure XML message as the POST payload. The server responds with a HTTP response with the Infrastructure XML acknowledgement message as the response payload.

5.4.3.1.1 SIF HTTPS Request Headers

There are some HTTP request and common headers that must be present in all SIF HTTPS messages sent by a client. These are: Content-Length, Content-Type and Host.

In addition to the preceding headers, a client may include a "Connection: close" header in the HTTP request if it wishes to close the current connection after receiving the response.

The client may include additional headers however the server must be able to successfully process POST requests that only contain the required headers.

5.4.3.1.2 SIF HTTPS Response Headers

The following HTTP response and common headers must be present in all SIF HTTPS responses messages sent by a server: Content-Length, Content-Type, Date and Server.

In addition to the above headers, a server may also include a "Connection: close" header in the HTTP response if it wishes to close the current connection after sending the response.

The server may include additional headers however the client must be able to successfully process Response notices that only contain the required headers.

5.4.4 SIF HTTP Transport

The SIF HTTP protocol is identical to the SIF HTTPS transport but does not include the TLS 1.0 security layer, which provides data encryption and authentication. An Agent or ZIS may implement the SIF HTTP transport but must implement the SIF HTTPS protocol. SIF Infrastructure, because of the sensitive data being exchanged by schools, only supports the HTTPS protocol.

6 Review on SIF Infrastructure, Architecture, Message Processing and Transport Layer

Figure 4 shows the layered infrastructure as it appears in the current SIF specification. The figure also presents where each layer is specified in the SIF documentation. The Binding to HTTPS (lower interoperability level) has been defined using a proprietary SIF solution. The top interoperability level (message format and message processing rules) has been defined using natural language in the SIF specification. Although these proposals provide interoperability at both levels, this section presents some recommendations (2 and 3) to use open and widely used technologies to solve the same interoperability problems.

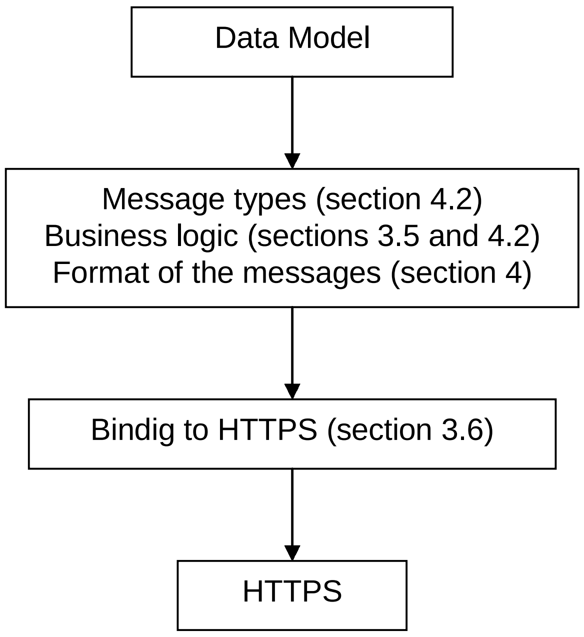

Data Model
Message types (section 4.2)
Business logic (sections 3.5 and 4.2)
Format of the messages (section 4)
Bindig to HTTPS (section 3.6)
HTTPS

Figure 2: Proposal by SIF

OASIS proposes to use JMS at the top level (see section 8.1). Interoperability at the lower levels remains on the same solutions as proposed by SIF (although there are some references to the use of SOAP). This alternative is shown in Figure 3 and implies the adoption of a concrete technology.

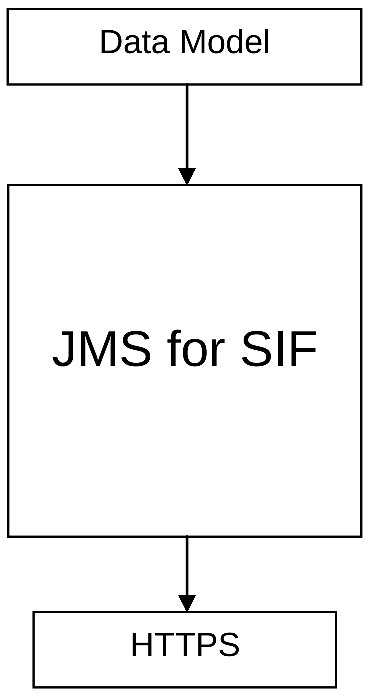


Figure 3: Proposal by OASIS

This CWA proposes the alternative shown in Figure 4. This maintains the same SIF general structure, although it proposes a concrete technology (SOAP) for the low interoperability layer and the formalisation of both message format and processing (using for example UML or BPEL4WS). Security management remains, as in the two previous alternatives, on the use of HTTPS. Also, the three proposals assume a separate data model over the message processing layer. The technology used to bind the conceptual data model is XML.

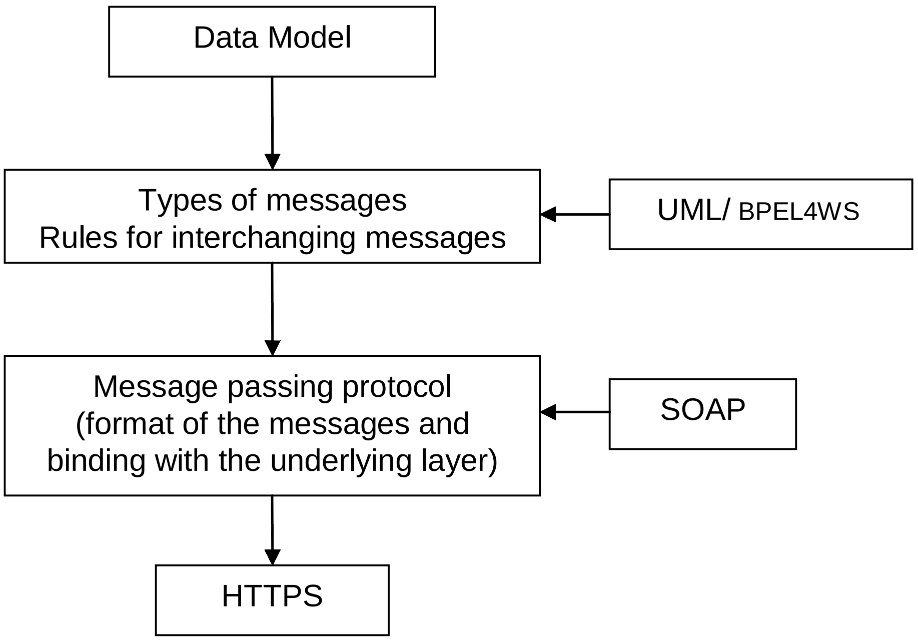

Data Model
Types of messages
Rules for interchanging messages
UML/ BPEL4WS
Message passing protocol
(format of the messages and
binding with the underlying layer)
SOAP
HTTPS

Figure 4: Proposal by CEN/ISSS WS-LT

Interoperability within a SIF environment is needed at two levels:

6.1 Interoperability at the top Level

At the top level we need a common Message Processing and Passing mechanism in order to guarantee that both SIF Agents and ZIS would process messages in the proper way according to a common protocol.

Message Processing and Passing business logic is specifed in SIF using natural language. Textual descriptions with some graphics are used to describe how message should be exchanged and how they should be processed by the different parties involved.

This CWA recommends at this level:

> *Formalise the SIF specification in so far as message passing and message processing is concerned, using a modelling mechanism as open and technology and platform independent as possible*.

The following two alternatives may be considered by SIF and further studied:

1. The Unified Modelling Language (UML) [5] is a graphical notation language which allows specifying and documentating software systems in a standardized way. UML comprises all the conceptual (e.g. business processes) and concrete (e.g. classes written in a specific programation language) aspects of a software system. UML offers different mechanisms for modelling of the statics but also the dynamics of systems. Among this mechanisms the most relevant are the sequence and interaction diagrams (activity diagrams). The first diagrams specify relations in a set of objects and the messages that are interchanged among them. Both the sequence of the messages and the structural organisation of the objects interchanged by this messages can be described. The activities diagram allows the specification of control flows. The negotiation process of a system may be documented and graphically modelled by the use of these two mechanisms at the same time.

2. The *Business Process Execution Language for Web Services,* BPEL4WS, [6] is being developed by a group of enterprises and institutions grouped in a Consortium: "Organization for the Advancement of

Structured Information Standards". In the same sense the W3C has created a group named “Web Services Choreography” whose main target is to define an official recommendation of a business process definition language. At the time this group has not published any draft and they only meet periodically. However the logical way forward will be to adopt BPEL4WS modified in some points. Obviously as a Web services technology it is designed for systems that work together with SOAP [7] as a messaging protocol and WSDL for defining the interfaces.

The above mentioned two alternatives provide a technology-independent mechanism to specify the business logic for message processing and passing. Nevertheless, the BPEL4WS has been designed with Web Services in mind, which leads to the second level where interoperability is required in a SIF environment as far as communication is concerned: message transportation.

The corresponding recommendation is **7.2**.

6.1.1 Application Programming Interfaces for Message Processing and Message Passing

The business logic for SIF message processing and passing needs to be formally specified as required in the previous sub-section. This must be included as a normative section in the SIF specification. There is no need to identify common APIs to encapsulate this logic, since the interoperability is guaranteed by the implementation of a common API, not by the API itself.

Nevertheless, the specification of a common interface may be proposed as a way to promote the development of reusable components, which, in turn, could be used by those developing SIF-compliant software. This type of interface definition should be included in a “best practice”-like document (no normative document).

Such a specification should be defined using an interface definition language as open and technology independent as possible. The OMG’s Interface Definition Language (IDL) [8] is the IDL proposed by this CEN/ISSS WS-LT for the short term to be analyzed. However, any other interface language should be appropriate enough provided it has no tie to any technology or implementation environment.

Once the Application Programming Interface has been specified at the conceptual level the next step should be to bind this conceptual interface to concrete interfaces for particular implementation environments and programming languages (e.g. Java, C++, etc.). Such bindings may be included in additional documents or appendices to the “SIF APIs best practice guidelines” proposed in this section.

The corresponding recommendation is **7.3**

6.2 Interoperability at the lower level

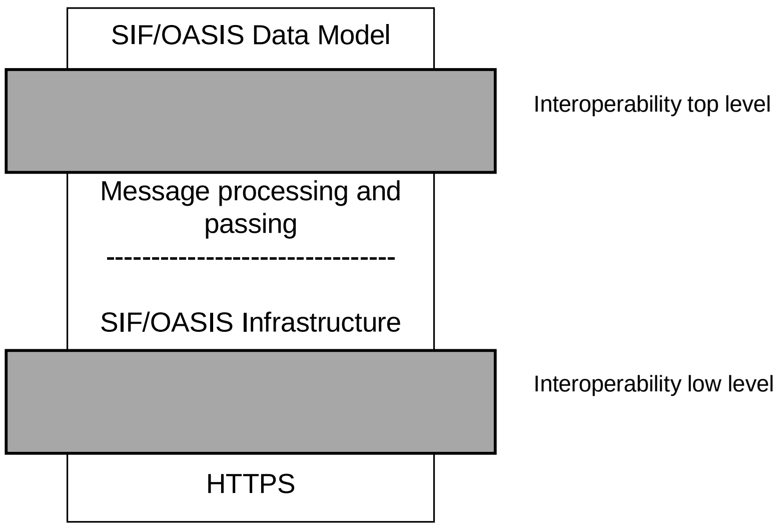
Currently, SIF specifies its own mechanism to encapsulate SIF messages using the HTTPS protocol, which, on the other hand is the mechanism used to provide the required security level. As explained in the SIF specification document (section 3.6) the Infrastructure messages are used by SIF to encapsulate and transfer the data objects. A client using the HTTPS protocol opens a connection to the server and sends a HTTP 1.1 POST request with the SIF Infrastructure XML message as the POST payload. The server responds with the Infrastructure XML acknowledgement message as the response payload. Each message must carry some headers (e.g. Content-Length or Content-Type).

Nevertheless, in the present situation, there exists a widely used technology that provides the same functionality: SOAP [7]. SOAP provides a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized, distributed environment using XML. SOAP does not itself define any application semantics such as a programming model or implementation specific semantics; rather it defines a simple mechanism for expressing application semantics by providing a modular packaging model and encoding mechanisms for encoding data within modules. This allows SOAP to be used in a large variety of systems ranging from messaging systems to RPC.

SOAP consists of three parts:

- The SOAP envelope construct defines an overall framework for expressing what is in a message; who should deal with it, and whether it is optional or mandatory.
- The SOAP encoding rules define a serialisation mechanism that can be used to exchange instances of application-defined data types.
- The SOAP RPC representation defines a convention that can be used to represent remote procedure calls and responses.

In fact, SIF references SOAP and proposed its study for further adoption. SOAP was not adopted by SIF to provide interoperability at this level simply because it was not available at the time SIF was firstly defined. A general recommendation by this CWA is to adopt SOAP as the protocol to use over HTTPS to exchange messages in a SIF environment (see section 7). In this way, interoperability will be covered at the two key levels, as needed in a peer-to-peer communication in a SIF environment (see Figure 5)


SIF/OASIS Data Model
Interoperability top level
Message processing and passing

SIF/OASIS Infrastructure
Interoperability low level
HTTPS

Figure 5: Levels of interoperability

The corresponding recommendation is **7.4**.

6.3 Organization of the SIF Specification

A deeper analysis of what is presented in section 3 of SIF specification [4] leads this CEN/ISSS WS-LT to recommend the re-organisation of the SIF specification in order to keep “Architecture” and “Infrastructure” clearly separated in the specification.

“Architecture” may be generally defined as the identification, organisation and distribution of those components that together will provide the SIF environment (e.g. Agents and ZIS). The conceptual procedures to communicate agents within a SIF environment (e.g. Request/Response or Publish/Subscriber models) also belong to the architectural description of SIF. Nevertheless, the actual implementation of these procedures (e.g. message format, message description, concrete message exchange, etc.) belongs to the “Infrastructure”. Generally speaking, those details that define the communication between to peers (in this case Agent and ZIS), including message processing, passing and definition and its lower level counterparts, should be included in the “Infrastructure” specification.

Defining what is “Architecture” and what is “Infrastructure” is a complex issue that always implies a certain degree of subjectivity. Nevertheless, there are several cases where both concepts are clearly mixed in the original SIF specification. For example, section 3 is devoted to the SIF architecture, whereas some of its subsections clearly deal with infrastructure:

- Section 3.5 that deals with Message Processing.
- Section 3.6 that corresponds with Infrastructure Transport Layer
- Sections 3.3.3 and 3.3.4 that manages the Data Provision: the Request and Response model and the Event Reporting: A Publish and Subscribe Model
- Section 3.4.5.2 that deals with Access Control List
- Section 3.4.5.6 that manages the Use of Selective Message Blocking (SMB) to Resolve Deadlocks

The following table analyses in detail the SIF specification giving a starting point for further discussion on where each section should belong to (“Infrastructure” or “Architecture”):

SIF Architecture Section	Section recommended by CEN/ISSS WS-LT
3.3.1	*Architecture* ☒ *Infrastructure* ☐ *Both* ☐
3.3.2 to 3.3.4	*Architecture* ☐ *Infrastructure* ☐ *Both* ☒

	High level to architecture and low level to infrastructure. To be further studied.
3.3.5	*Architecture* ☐ *Infrastructure* ☒ *Both* ☐
3.3.6	*Architecture* ☐ *Infrastructure* ☐ *Both* ☒ High level to architecture and low level to infrastructure. To be further studied.
3.4.1 to 3.4.3	*Architecture* ☒ *Infrastructure* ☐ *Both* ☐

3.4.4 and 3.4.5	*Architecture* ☐ *Infrastructure* ☐ *Both* ☒ High level to architecture and low level to infrastructure. To be further studied.
3.5 and 3.6	*Architecture* ☐ *Infrastructure* ☒ *Both* ☐

Table 2: SIF Architecture and Infrastructure

The CEN/ISSS WS-LT proposes analysing the multilinguality issue at the data model and avoiding any reference to it at the infrastructure or architectural levels.

The corresponding recommendation is **7.1**.

7 CEN/ISSS WS-LT Recommendations

This section includes five concrete recommendations to the SIF community and other SIF-based initiatives regarding Infrastructure, Architecture, Message Processing and Transport Layer.

7.1 Recommendation 1

The need	Defining what is “Architecture” and what is “Infrastructure” is a complex issue that always implies a certain degree of subjectivity. Nevertheless, there are several cases where both concepts are clearly mixed in the original SIF specification (see section 3.3).
Action	Re-organize the SIF specification, in particular the section dealing with Architecture, in order clearly to separate the specification on the SIF “Architecture” and the SIF “Infrastructure”. Section 6.3 presents an initial proposal that may be used as a starting point.

7.2 Recommendation 2

The need	Message Processing and Passing business logic is specifed in SIF using natural language. Textual descriptions with some graphics are used to describe how message should be exchanged and how they should be processed by the different parties involved. A formalisation of this description is needed.
Action	To formally specify the SIF message model including message format and message processing rules. For this, the use of UML or BPEL4WS (see section 6.1) may be considered.

7.3 Recommendation 3

The need

The specification of a common API would promote the development of reusable components, which, in turn, could be used by those developing SIF-compliant software. This API would encapsulate the business logic for message processing and passing, which should be formally defined according to Recommendation 7.2.

Action

To specify a common interface encapsulating the business logic for message processing and passing. The OMG’s Interface Definition Language (IDL) [8] is a technology-independent language for interface definition that may be considered as an initial approach. This type of interface definition should be included in a “best practice”-like document (no normative document).

Once the Application Programming Interface has been specified at the conceptual level the next step should be to bind this conceptual interface to concrete interfaces for particular implementation environments and programming languages (e.g. Java, C++, etc.). Such bindings may be included in additional documents or appendixes to the “SIF APIs best practice guidelines” proposed in this section.

7.4 Recommendation 4

The need	The SIF specification defines in its section 3.6 its own mechanism to encapsulate SIF messages using the HTTPS protocol. Currently, there exists an open and widely used technology that supports the same functionality: SOAP. The use of SOAP would ease the implementation of SIF-compliant tools.
Action	To adopt the use of W3C's SOAP as the mechanism to encapsulate and exchange messages over HTTPS in the SIF infrastructure model.

7.5 Recommendation 5

The need	The SIF specification limits queries to a subset of logical conditions (at present, only equality is defined as a test mechanism). A wider mechanism, including other logical conditions, would improve object queries in SIF. The rationale for this recommendation in included in section 8.3.3.4
Action	To define alternative logical operations to the equality and a more elaborated language for SIF objects queries.

8 Appendixes

8.1 Appendix A: SIF Message Passing by OASIS

8.1.1 Messaging models for School Architectures

This document gives a general overview of Messaging and discusses SIF and some of its limitations with regards to the Oasis project. From an architectural standpoint, Oasis needs to create a communications infrastructure that is very flexible and can fit the initial architectural needs and any future extensions.

SIF, or a European version of SIF, should be considered as a framework for interoperability between administrative applications, while other more general-purpose communications models, such as a general Messaging Service, or synchronous communication technology, for example SOAP, should also be put in place as part of the general communications infrastructure.

This document will discuss general messaging, SIF, and the Java Messaging service.

The SIF implementation that OASIS implements is obviously not the whole architecture provided by the “School Server” and “Zone Management Server” architecture. The message passing architecture presented and described in the following pages only applies to the SIF implementation. All other components do not use specific messaging models (and especially not an asynchronous model) other than the market standards. The services outside the SIF may want to adopt in the future on the asynchronous model, and this seems to be a trend in similar projects outside Oasis, where the “federated services oriented” architecture is being adopted.

8.1.2 Overview Messaging – AKA MOM

There are many mechanisms to distributed applications, including Berkley Sockets, Java RMI, CORBA, and others. SOAP (serial object access protocol) is a Web services method and service invocation that is becoming increasingly important.

However these methods rely on mostly synchronous connections (Remote Procedure Calls or RPCs).

Messaging, or Message Oriented Middleware (MOM), in contrast generally does *not* rely solely on synchronous connections, and introduces the concept of asynchronous messages (with queuing) along with guaranteed delivery of the messages, and new Publish/Subscribe paradigms. In essence, a messaging system allows separate, uncoupled applications reliably to communicate asynchronously. The messaging system architecture generally replaces the client/server model with a peer-to-peer relationship between individual components, where each peer can send and receive messages to and from other peers.

Messaging systems are used to build highly reliable, scalable, and flexible distributed applications including Web and Educational applications. Messaging systems attempt to solve synchronization, reliability, scalability, and security.

Messaging is a different programming paradigm from synchronous, or connection oriented methods. Synchronous connections are easily mapped to function-oriented request/reply applications with application programming interfaces (APIs), while message-oriented middleware's asynchronous, or non-blocking, interaction is extremely useful for event-oriented applications.

In contrast, the RPC application model transfers control over the network and to the remote node. While the remote node is processing the request, the user waits for the response and cannot do anything else in that application. If the user is connected via a Web browser, he or she cannot use the browser until the response comes back. The application is in effect blocked.

Event-oriented applications do not pass control over the network, they just send messages that represent work (transactions) and receive answers back. In the meantime, the application can get other things done. It could even send out more requests to different networked servers and allow all its responses to come back individually without having to sequence them one after another. This parallelism can make more complex applications respond more quickly just by doing more than one thing at once.

Given current J2EE implementations, it is possible to build multi-threaded Java servlets that act in an asynchronous interaction with updates occurring in the background. Building this however increases the application development effort since more programming must be done to handle multiple simultaneous threads in the application. Most Messaging products, in contrast, often offer synchronous behaviour so that applications needing that level of responsiveness can get it.

8.1.3 Improved Load Characteristics – Messaging and MOMs

Message-oriented solutions are not typically client/server session based. Unlike synchronous connection oriented data access solutions, messaging solutions include session-like information in each message, and do not require setup and teardown mechanics. The application just sends a message. They are typically connectionless, rather than connection-oriented.

This can lead to improved load characteristics. Session-based solutions, and data access products, often require too much overhead on the server to handle conversations with a large number of clients. All that's important is the number of transactions per second, not how many total users that represents.

Because of improved load handling characteristics, implementing with a MOM or messaging technology becomes a viable solution, especially when implemented in restricted bandwidth WAN or disconnected scenarios where session-oriented solutions are expensive.

8.1.3.1 Limitations of and application areas of SIF

The primary focus of SIF is in five areas of School Administration:

- Student Administration
- Human Resources and Financials
- Library
- Food Service
- Transportation

Therefore, SIF attempts to solve the interoperability between the applications specifically to solve these issues:

- Administrative applications and their data are separate and isolated from each other
- Data is entered in each application separately and redundantly, and thus not shared
- Support costs are increased due to the isolation of the administrative applications and data
- Data reporting is not integrated and costly
- Data is just not available to school officials in a consistent form

SIF is implemented as a Message Oriented Middleware (MOM) that has standardised on specific types of messages, and transport protocol in the Administrative space. All of the messages, transport, and Application Programming Interfaces (APIs) are defined. SIF specifies a Zone Integration Server (ZIS) that acts as the message router and queues messages for delivery. SIF provides an API for creation of Agents that can, handle all communication with the ZIS. SIF is not a general purpose Messaging Service; however it can be used in conjunction with others as long as the Agents are programmed that way.

An example of a typical SIF implementation would be to connect products from different vendors together within a single school. These applications might include a student information system (SIS) application, a food service program, and a library automation application. Each of these applications will need to have a vendor provided interface program, for communication with the ZIS, called an "Agent." Agents are custom applications that act as the go between for the proprietary application and the SIF framework.

The data that can be exchanged in SIF is pre-defined using XML data objects. These XML objects define the semantics of information that can be managed by the applications. As an example the following are defined in the SIF:

- StudentSchoolEnrollment,
- StudentPersonal,
- StaffPersonal
- Address

All SIF interchanges must use these pre-defined data objects and agents must exist to transform internal proprietary data formats into the pre-defined objects and interface with the ZIS for the exchange of data. Messages that do not fit the SIF standard will generate an error.

8.1.3.2 SIF XML data object

It is important to note that all of the XML objects are fully specified and are thus fixed in the standard. An example from the standard:

Address

This element contains information about the address. The Address element always occurs within another object such as StudentPersonal/StudentAddress.

Element	Attrib	Char	SP/Ex ID	SP/Ex Codes	Description
Address		O			This element contains information about the address. The address element could be included in a student address, school address or contact address.

Element	Attrib	Char	SP/Ex ID	SP/Ex Codes	Description
	Type	R		**Code & Description** AC -City and State CC-Country CI-City CY-County/Parish DR-District of Residence F-Current Address L-Local Address M-Mailing Address O-Office Address P-Permanent Address PT-3 digit Canadian Postal Code PU-6 digit Canadian Postal Code RE-Regional Education Service Agency SB-Suburban SD-School District SH-School Campus Code SP-State/Province SS-School TN-Township UR-Urban ZZ-Mutually Defined	SPEEDE-ExPRESS code that defines the location of the address.
Street		M			The street element is a complex element and breaks the street down into several parts.
Street/Line1		M			Address line 1.
Street/Line2		O			Address line 2.
Street/Line3		O			Address line 3.
Street/Complex		O			Name of the complex.
Street/Street		O			The number of the

Element	Attrib	Char	SP/Ex ID	SP/Ex Codes	Description
Number					street.
Street/Street Prefix		O			Street prefix like NE
Street/Street Name		O			The name of the street.
Street/StreetType		O			The type of street. For example, Lane, Blvd., Ave etc.
Street/Street Suffix		O			Street suffix like SW.
Street/AptType		O			Type of apartment, for example, Suite.
Street/AptNum Prefix		O			Apartment number prefix.
Street/Apt Number		O			The number of the apartment.
Street/AptNum Suffix		O			Apartment number suffix.
City		M			The city part of the address.
County		O			The county part of the address.
StatePr		M			The state or province.
	Code	R	N402-156	See Appendix E.	The state code.
Country		M			
	Code	R	N404-26	See Appendix E.	The country code.
PostalCode		M			The ZIP/Postal code.
GridLocation		O			The location of the address. For a description of this element, see the GridLocation specification.

Table 3: Address Element

```
<Address Type="M">
    <Street>
        <Line1>1 IBM Plaza</Line1>
        <Line2>Suite 2000</Line2>
        <Line3>Chicago, IL 60611</Line3>
        <StreetNumber>1</StreetNumber>
        <StreetName>IBM</StreetName>
        <StreetType>Plaza</StreetType>
        <AptType>Suite</AptType>
        <AptNumber>2000</AptNumber>
    </Street>
    <City>Chicago</City>
    <County>Cook</County>
    <StatePr Code="IL"/>
    <Country Code="US"/>
    <PostalCode>60611</PostalCode>
    <GridLocation>
        <Latitude>41.7699657</Latitude>
        <Longitude>79.548445</Longitude >
    </GridLocation >
</Address>
```

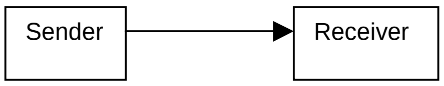
8.1.3.3 Final note on SIF

As a messaging model, SIF is a very good implementation that fits a specific need in North America. A European version is needed to fit specific European requirements.

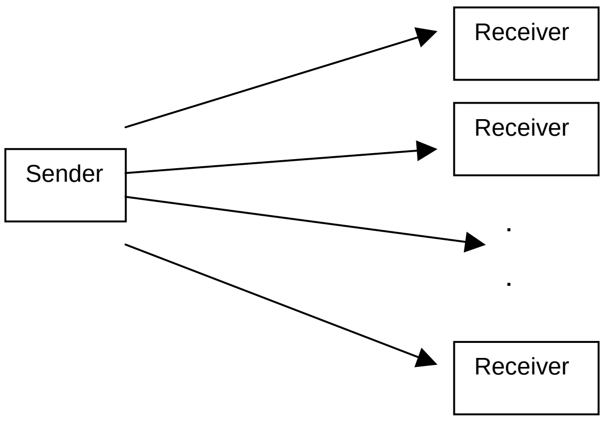
Oasis should consider using SIF as a model for Messaging implementation.

8.1.4 Messaging three basic patterns

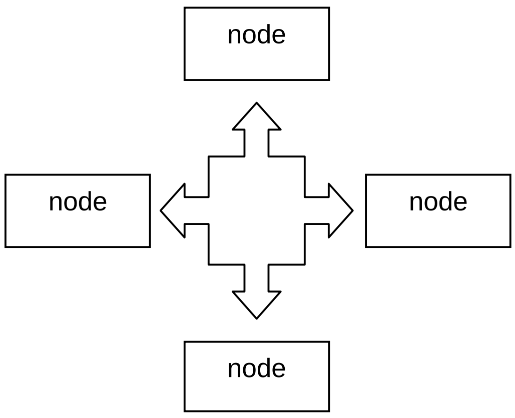
1-to-1, point-to-point



1-to-many, can be publish/subscribe

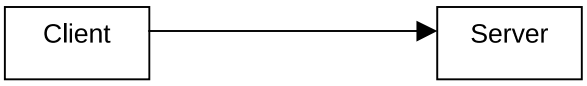


Many-to-many

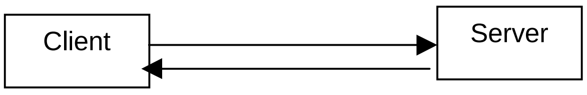


8.1.4.1 Messaging has 4 basic interaction patterns:

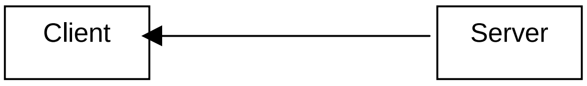
One-way



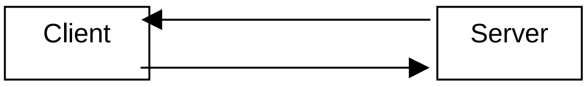
Request/response



Notification



Notification/response



Queues can be employed at either end of a message, at the client, or at the server, or both, or neither. Scheduling can be employed to deliver messages, or of course they can be "real-time" in nature.

8.1.5 Java Message Service (JMS)

The Java Message Service provides a consistent API set that gives developers access to the common features of many messaging system products. Examples of other messaging products include IBM's MQ series and Tibco.

8.1.5.1 Message System Types

Two messaging systems models are in common use:

8.1.5.1.1 Publish/Subscribe-

A publish/subscribe (pub/sub) messaging system supports an event driven model where information consumers and producers participate in the transmission of messages. Producers "publish" events, while consumers "subscribe" to events of interest, and consume the events. Producers associate messages with a specific topic, and the messaging system routes messages to consumers based on the topics the consumers register interest in. An example might be the publishing (broadcast) of a news activity of interest to “8th grade” teachers. Only those teachers would receive the message and view the activity message.

8.1.5.1.2 Point-To-Point-

In point to point messaging systems, messages are routed to an individual consumer (could be an application), which maintains a queue of "incoming" messages. Messaging applications send messages to specified queue, and clients retrieve messages from a queue. The queue maintains persistence of the messages until they are delivered, thus if the sender or the receiver machines go down, the messages are still delivered to the individual consumer when they come back on line.

Frequently vendors will support either point-to-point, or publish/subscribe messaging models, or both.

8.1.5.2 The Java™ Message Service (JMS)

Java Message Service, part of the J2EE (Java 2 Enterprise Edition) suite, provides standard APIs that Java developers can use to access the common features of enterprise message systems. JMS supports the publish/subscribe and point-to-point models and allows the creation of message types consisting of arbitrary Java objects.

JMS can reside in the Application Integration tier of a Web Service, thus allowing a mechanism for either point-to-point, or publish/subscribe messages to other applications. It is also at the Application Integration Tier that SIF agents would reside, a ZIS must also sit some where on the network to coordinate messages for the Administrative application Agents.

8.1.5.3 Design Goals

A fundamental design goal of JMS is to provide a consistent set of interfaces that messaging system clients can use independent of the underlying message system provider.

This way, not only is client application portable across machine architectures and operating systems, it is also portable across messaging products. Client applications written to JMS will work without modification on all JMS compliant messaging systems. (This can be compared to the independence of Enterprise Java Beans™ components on the underlying middleware where the components run.)

JMS was also designed to:

- Minimise the effort required by messaging system providers to implement the JMS APIs for their products.
- Provide most of the functionality of common messaging systems.

Many messaging system vendors have implemented JMS for their products, allowing Java access to the functionality of their systems. JMS Clients Can Use Java Facilities. JMS clients, since they are based on Java, can make use of existing Java APIs such as JDBC™ for database access, the JavaBeans™ components model, JNDI for naming services, JTA for client transaction control, or any of the J2SE™ and J2EE™ APIs for enterprise application services.

8.1.5.4 JMS Details

What is a Message?

In messaging systems the point of communication between applications is the message itself, so a developer using JMS must understand messages.. Although the definition of a message varies greatly between messaging systems, JMS provides a unified means of describing and accessing messages. A JMS message consists of three parts:

8.1.5.4.1 Message header

For message identification. For example, the header is used to determine if a given message is appropriate for a "subscriber"

8.1.5.4.2 Properties

For application-specific, provider-specific, and optional header fields

8.1.5.4.3 Body

Holds the content of the message. Several formats are supported, including TextMessages, which wrap a simple String; and ObjectMessages, that wrap arbitrary Java objects (which must be serializable). Other formats are supported as well.

8.1.5.4.3.1 TextMessages

A TextMessage wraps a simple String object. This is useful in situations where only strings are being passed. It is expected that many messaging systems will be based on XML, and TextMessages are a natural container for these. Creation of a TextMessage object is simple, as these two lines of code indicate:

```
TextMessage message =
    session.createMessage();
message.setText("hello world");
```

(We'll see the "session" object in the next section.)

A TextMessage created in this way is ready to be published to a messaging system.

8.1.5.4.3.2 ObjectMessages

An ObjectMessage, as its name implies, is a message wrapping a Java object. Any serialisable Java object can be used as an ObjectMessage. If multiple objects must be transmitted in a single message, then a Collection object (such as a List or a Set) containing several serialisable objects can be used.

This is how an Object message is created:

```
ObjectMessage message = session.createObjectMessage();
message.setObject(myObject);
```

Note on JNDI

JMS, like many of the J2EE APIs, makes use of JNDI (Java Naming and Directory Interface) to discover needed resources. A detailed discussion of JNDI is beyond the scope of this document, but more information can be found at Java Naming and Directory Interface^{TM} (JNDI)

8.1.5.5 Building a JMS Client

A typical JMS client can be built following these basic steps:

1. Create a connection to the messaging system provider
2. Create sessions, for sending and receiving messages
3. Create MessageProducers and MessageConsumers to create or receive messages

Once these steps have been performed, a message-producing client will create messages and publish them to topics, while a message-consuming client will listen for messages associated with a topic, and consume them as they arrive.

To illustrate in detail how this works, we examine a typical message producer, used to publish messages to a topic in a pub/sub messaging system. Note that all exception handing code has been omitted for clarity.

8.1.5.5.1 Create a Connection

A Connection provides the client access to the underlying messaging system, and performs resource allocation and management. Connections are created using a ConnectionFactory, which is typically located using JNDI.

This code illustrates the steps involved in creating a connection:

```
Context messaging = new InitialContext();

// get JNDI context

TopicConnectionFactory topicConnectionFactory =

  (TopicConnectionFactory)

    messaging.lookup("TopicConnectionFactory");

TopicConnection topicConnection =

 topicConnectionFactory.createTopicConnection();
```

8.1.5.5.2 Create Sessions

Sessions are lightweight JMS objects which provide a context for producing and consuming messages. Sessions are used to build message producers and message consumers, as well as to build the messages themselves.

```
TopicSession session =

  topicConnection.createTopicSession(false,
                        Session.CLIENT_ACKNOWLEDGE);
```

The two parameters to createTopicSession() control transactions and message acknowledgment.

8.1.5.5.3 Locate a Topic

A topic (also known as a subject, group, or channel) is located through JNDI. The topic identifies the messages being sent or received. In pub/sub systems, the subscribers subscribe to a given topic, while the publishers associate topics with the messages they publish.

Here we create a topic called "WeatherData."

```
Topic weatherTopic = messaging.lookup("WeatherData");
```

Start the Connection

During the above initialisation steps, message flow is inhibited to prevent unpredictable behaviour during initialisation. Once initialisation is complete, the connection must be told to initiate the message flow.

```
topicConnection.start();
```

8.1.5.5.4 Create a Message Producer

In the publish/subscribe domain, a producer publishes messages to a given topic. This code shows the creation of a publisher and the subsequent building and publishing of a simple text message.

```
TopicPublisher publisher =
    session.createPublisher(weatherData);
TextMessage message = session.createMessage();
message.setText("temperature: 35 degrees");

publisher.publish(message);
```

A similar process is followed to create subscribers, as well as JMS clients for point-to-point systems.

For more information and documentation on the Java Message Service, see:

Java^{TM} Message Service Documentation (http://java.sun.com/products/jms)

8.1.6 Suitability for Europe:

JMS is a strategic technology for J2EE. JMS will work in concert with other technologies to provide reliable, asynchronous communication between components in a distributed computing environment. As such, JMS is an ideal solution for generalized messaging. It is complementary with SIF.

In view of the most advanced messaging architectures supported by the European Commission through FP5 funded projects, there is a trend towards a "federated services oriented architecture".

The architecture of the SIF message passing structure is the best possible way to achieve this goal, as it allows for any later adopted standard service to be included in this message passing structure. The asynchronous message passing is the only one of its kind in the marketplace which has this ability, whatever the new service.

8.1.6.1 To be localised:

Java and JMS support localised languages.

No other localisation is requested at the message passing level.

8.1.6.2 Not relevant:

Java can accommodate many different technologies and supports current development methodologies and multiple platforms. SIF implementations support Message Transport Service API, which JMS can adhere too.

8.1.6.3 In conflict:

Currently no real conflicts, as long as the components to be adopted comply with the J2EE/JMS framework.

8.1.7 Compatibility with European developments:

JMS is part of JAVA/J2EE environment, as such it is quite complementary with developments happening with SIF.

8.1.7.1 Complementary

Agents and ZIS connections can be built with JAVA /J2EE Enterprise Beans, etc. and use JMS messaging. Connections will have to be made between non-SIF applications and agents, so JMS can work here. In some cases a generalised messaging service can exist outside the "standardised SIF" messaging model.

8.1.7.2 In conflict:

No Conflict.

8.1.7.2.1 Negotiate with the SIF a generalisation

JMS will not affect SIF generalisation. It might be useful for interim European SIF agent messages.

8.1.7.2.2 Change SIF version for Europe

The SIF Specification currently defines 19 Data Objects, with an additional 40 Data Objects in the Draft mode, and more to be defined as the Specification evolves. By agreeing on these definitions, SIF makes it possible for software programs built on different platforms and with different database designs to share data.

8.1.8 Recommendations:

8.1.8.1 List of the Parts of SIF to be adopted

SIF Specification v1.0r1

- The SIF Specification is based on the World Wide Web Consortium (W3C) endorsed Extensible Markup Language (XML) which is not linked to a specific operating system or platform. XML defines common data formats and rules of interaction and architecture.
- Event Reporting is a Publish/Subscribe Model.
- Events are published/reported by each application and are received by other applications that have subscribed to them.
- DataProvision is a Request/Response Model. An application agent may register itself to the Zone Integration Server (ZIS) as a provider of certain data objects. This allows the ZIS to satisfy a query request from an agent to locate the provider of a data object. The ZIS acts as the intermediary, forwarding the request to the agent, receiving the response, and then forwarding the response back to the original requester.
- Messages are securely encrypted using HTTPS. HTTPS is the most commonly used secure method of exchanging data among web browsers.
- Agents are authenticated by the ZIS before messages are passed.
- Message processing includes message validation and message identification. Guaranteed message delivery.

8.1.8.2 List of the Parts of SIF to be left out of the European standard

None, until shown to be of no use.

8.1.8.3 List of the Parts of SIF to be negotiated as a change in the World standard

As SIF started in the U.S.A., Internationalisation and Localisation need to be confirmed for SIF Europe. Extensions to existing standards will have to take place.

Localisation to European data exchange models need to occur.

8.1.9 Other comments

The current draft of SIF is sufficient to identify Data Objects for Europe. A European SIF Connect-a-thon would be useful to test U.S. SIF vs. European applications and vendors. Creation of a SIF laboratory for trial of ZIS and Agents from various vendors should occur.

The creation of a small number of demonstrator sites should be created. Examples in the U.S.A are:

SIF SHOWCASE SITES

SIF showcase sites demonstrate how we work with educators to ensure maximum benefits from technology investments in real world applications.

Peoria, AZ: Copperwood Elementary Staff Facilitates Student Services While Reducing Data Entry Redundancy and Errors

Peoria, Arizona

Ocoee, FL: Middle School Aims to Work Smarter with SIF Through Data Sharing

Ocoee, Florida

At Anoka-Hennepin ISD, "It's Getting Better All the Time"

St. Paul/Minneapolis, MN

Upper Dauphin: Standing on the Leading Edge with SIF

Lykens, Pennsylvania

JMS, or other messaging systems can bridge Web applications to SIF. Eventually, message APIs should be subsumed into SIF and form part of the general services provided by SIF implementations. The current SIF does not support “Web Services” as defined by SOAP, WSDL, JMS, etc. or XMS schema. This should be addressed in future versions with European input.

8.2 Appendix B: SIF Architecture by OASIS

8.2.1 Introduction

8.2.1.1 Scope

The scope of this document is Work Package 2000 of the OASIS project.

8.2.1.2 Purpose

The purpose of this document is to provide a report on the SIF Architecture and its relevance to the OASIS project. The document is organized as follows: first, a summary of architectural issues of the SIF specification is provided; then, the relationship between SIF and European educational developments and initiatives (OASIS among them) is reviewed; finally, the conclusions are presented.

8.2.1.3 Glossary

• CORBA	Common Object Request Broker Architecture,
• CSCW	Computer Supported Cooperative Work.
• ETB	European Treasury Browser.
• EU	European Union.
• EUN	EUropean schoolNet.
• HTTP	HyperText Transfer Protocol.
• HTTPS	HyperText Transfer Protocol, Secure.
• NNTP	Network News Transfer Protocol.
• OASIS	Open Architecture and Schools In Society.
• OCSL	Open Code Software Library.
• PC	Personal Computer.
• RMI	Remote Method Invocation.
• SIF	Schools Interoperability Framework.
• SMTP	Simple Mail Transfer Protocol.
• USA	United States of America.
• VWE	Virtual Workspace Environment.
• XML	eXpress Markup Language.
• ZIS	Zone Integration Server.
• ZMS	Zone Management Server.

8.2.2 Sources

1. Schools Interoperability Framework. Implementation Specification. Version 1.0 Revision1. August, 2001. http://www.sifinfo.org
2. SIF Standard for K-12. Where do we go from here? Ron Kleinman, 21/1/2002.
3. EUN Multimedia MM1010. Deliverable: EUN Architecture and Framework D06.02.01. May, 2000.
4. European Treasury Browser. Deliverable D9.1: WP9 Architecture of the Metadata Networking Infrastructure. October 2000.
5. OASIS Shared Cost RTD Project. Annex 1. Description of Work. 19/7/2001.
6. Extensible Markup Language (XML) 1.0 (Second Edition) W3C Recommendation, 6th October 2000. http://www.w3.org/TR/REC-xml
7. WP3000 Detailed Work Plan. DIT Document. Ángel Fernández del Campo. 15th Jan 2002.

8.2.3 Procedure in analysis

In this section the SIF Architecture is reviewed in order to identify its main characteristics and the relevant issues for the OASIS project.

8.2.3.1 Introduction to SIF

Schools Interoperability Framework (SIF) is the place where companies and educators will be able to participate in the development of specifications that will ensure interoperability in the K-12 instructional and administrative environment. The aim of a SIF implementation is to provide applications related to schools or education with effective means of sharing data. In the USA, where this specification has been developed, some example areas of these applications are: student information systems, food services, transportation services, library applications, etc.

8.2.3.2 SIF architecture

8.2.3.2.1 Basics

Although there are many variations of SIF topographies, the common feature is that <u>a number of applications wish to share data</u>. SIF groups applications wishing to share data together into a logical entity which is called *Zone*. All SIF implementations regardless of their complexity consist of one or more applications with their associated interface *agents* all being managed by a *Zone Integration Server* (ZIS). Thus, applications interact by means of integration *agents* which communicate with the ZIS. Agents are typically provided by the application vendor. An agent <u>never</u> talks to another directly. All communications between agents are managed by the ZIS.

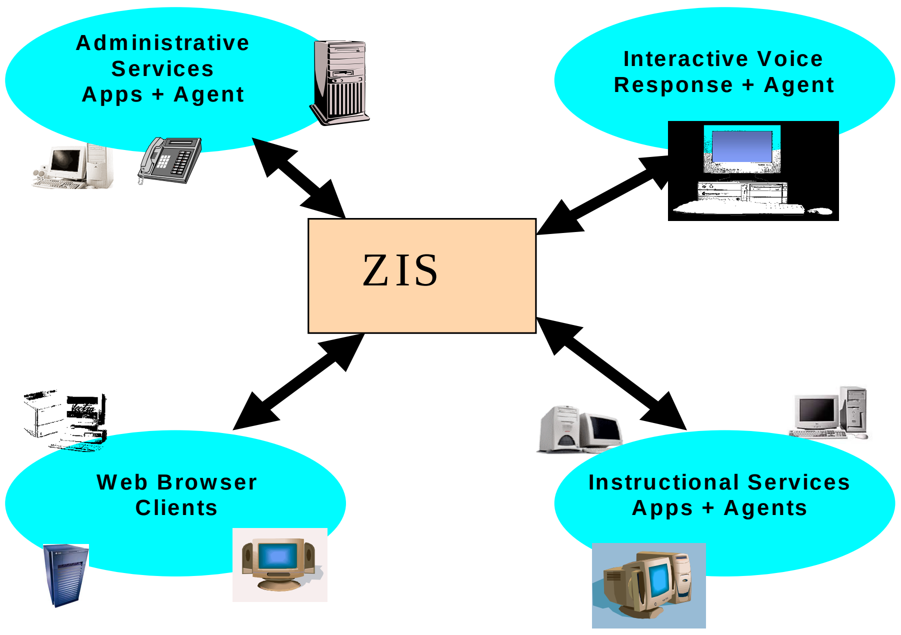

Administrative Services Apps + Agent
Interactive Voice Response + Agent
ZIS
Web Browser Clients
Instructional Services Apps + Agents

Figure 6: SIF Architecture

A Zone Integration Server is a program that provides integration services to all the agents registered with it so that they can provide data, subscribe to events, publish events, request data, and respond to requests. It is responsible for all access control and routing within the system.

There are no predefined sizes for zones, so a zone can be as large or small as required in order to meet the needs of the customer. The size of the zone is flexible and could consist of a single building, school, a small group of schools, a district, etc.

A SIF implementation must enable different applications to exchange data <u>efficiently</u>, <u>reliably</u>, and <u>securely</u> regardless of what platforms are hosting the applications.

8.2.3.2.2 Naming

SIF requires that each agent and ZIS be identified with a distinct case-sensitive identifier that is unique within the zone where the agent/ZIS belongs.

8.2.3.2.3 Data representation: XML

Data that can be exchanged in SIF is defined as a series of data *objects* defined using XML.

8.2.3.2.4 Retrieval of information: Request/Response

Applications can query information from specific SIF data objects in a Request/Response fashion. The requester's agent sends a message to the ZIS, which will forward the request to the destination agent. If the destination of the requester is not specified, the ZIS will send the request to the *provider* for the object being queried. Thus, the provider of an object is the default responder application for requests to that object.

To act as a provider, an application must notify this to the ZIS. There is a *single* provider per object per zone, but there may be *multiple* responders for a given object in a zone.

8.2.3.2.5 Data updates: Publish/Subscribe

Applications can propagate data updates by making agents publish *events* that are reported to other agents interested in data updates. Events are sent to the ZIS by the generating agent, and the ZIS is responsible that event to the interested parties without any further communication to the event originator. Defined events are: “Add”, “Update”, “Delete”.

In order for an application to receive an event, *subscriptions* for the SIF data objects of interest must be entered at the ZIS. Thus, event reporting is handled with a Publisher/Subscribe model.

8.2.3.2.6 Communication: asynchronous model

The reliability requirement implies that when an application sends out a message to a destined receiver, the delivery of the message is guaranteed. This requirement also implies that messages sent by an application will arrive at the receiver in the same order as they are sent and that each message is delivered once and only once.

To ensure scalability and reliability, all communications between agents and ZIS is *asynchronous*. A ZIS or agent cannot be assured that it will get an immediate response to a message it submitted. When an agent is not connected to the SIF environment, other agents can still send messages to it, as the ZIS will deliver them as soon as the agent is available. A message queue solution is suggested in the specification, and should feature a high availability and reliability implementation (e.g. permanent storage, so that messages cannot be lost).

Two models for delivering messages to an agent are defined:

- *Push* model refers to the action by a ZIS to actively deliver messages to an agent without the agent having to initiate contact with the ZIS.
- *Pull* model refers to the action by an agent to explicitly request a single message from the ZIS.

8.2.3.2.7 Registration

Before an application can use the services of the ZIS, the application must *register* itself with the ZIS. During the registration process, information about the SIF specification version supported by the agent and the maximum supported size of an incoming packet from the ZIS is also communicated to the ZIS. Also, the way the ZIS may contact the agent (i.e. Push mode support) is noted during registration.

8.2.3.2.8 Networking

The figure below shows the protocol stack for carrying SIF XML messages. Support for HTTPS transport is **required** for SIF agents and ZIS. But other alternatives are possible: HTTP is suggested in the SIF specification, but implementers are free to choose other transports. For instance, Microsoft provides a ZIS based on their BizTalk Server, which can use HTTPS/HTTP, SMTP, local file copying, and Microsoft Message Queue for transporting SIF messages.

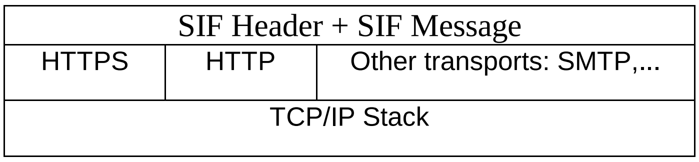


Figure 7: Communication stack in SIF

8.2.3.2.9 Security

Due to the sensitive nature of many of the interchanged data, a security model is required. SIF security model centres around three areas: encryption, authentication and access control.

- Encryption provides the mechanism to ensure that only the sender and receiver of a message can view the message contents. All agents and ZIS implementations must support the SIF HTTPS protocol, which provides a secure transport, with encryption of the data being exchanged. Other transports may be used, and, if so, it is important to know if they are secure to avoid exposing sensitive data. The ultimate responsibility for guaranteeing the security of data lies with the originating agent or zone administrators. Several levels of encryption are defined: no encryption, encryption with symmetric key length of 40, 56, 80 and 128 bits.
- Authentication. The aim of authentication is to provide a means to ensure that the author of a message is the actual author. Another requirement is message integrity. Authentication support is optional, but highly recommended. The SIF HTTPS protocol supports authentication between an agent and a ZIS. If authentication is enabled, a message receiver can trust the SIF HTTPS implementation to verify that the message in its entirety comes from the claimed sender.
- Access control. Its target is to determine which entities are allowed particular types of access to each data object. The current (1.0r1) SIF specification relies on some implementation of this functionality. The specification defines an Access Control List behaviour, but does not specify any standard for implementation of it. Access Control should deal with agent permissions associated to provision, subscription, request, response, and events. A ZIS will exhibit behaviour with regard to the ACL that could be perceived by an Agent as if a virtual table exists defining the following information:

Field	Comments
Agent name	The unique ID for an Agent (provided as the Source ID in a SIF_Register message)
Object name	The Object being manipulated (e.g., StudentPersonal, etc.)
Provide	Can this Agent register as the provider for this object?
Subscribe	Can this Agent register as a subscriber for this object?
Report “Add” event	Can this Agent report "Add" events for this object?
Report “Update” event	Can the Agent report "Update" events for this object?
Report “Delete” event	Can the Agent report "Delete" events for this object?
Request object	Can this Agent request information for this object?

Respond to request for object	Can this Agent respond to a request for this object?

Table 4: Access control in SIF

8.2.3.3 Limitations of SIF version 1.0r1

8.2.3.3.1 Consistency problems

The current SIF specification (1.0r1) allows any application (with the suitable access control rights) to publish an event declaring that some data have changed, even though this application is not the provider for the data object. But there is no requirement for the provider to actually make the changes. Thus, there are potential synchronization and consistency problems. The on-going version (1.5) provides a mechanism to solve this problem, basically the differentiation between proposed events (which any application can do, but they are received only by the provider) and published events (which the provider does, after the changes have actually been done).

8.2.3.3.2 Zone architecture limitations.

Zones, according to spec1.0r1, sec. 3.3.1, can be as large or small as required to meet the needs of the customer. In sec. 3.4.1 it is suggested that zones are typically defined according to physical boundaries, and that scalability, security and manageability requirements also determine the definition of zones. At present a SIF zone is supported by a single ZIS, and zones are usually defined based on physical and management requirements.

In addition, at most one application can behave as a provider of an object.

In USA there are several cases where this model does not fit well:

- Different organization of the involved applications: for instance, several schools could share the same transportation services, but each one has its own student information systems.
- Schools with sub-units, each one with their own applications.

At present there is on-going work in future SIF releases to cope with this problem. SIF v2.0 will free the SIF zone from the “single provider for an object” limitation. Two main techniques have been proposed:

- SubZones. Its purpose is to provide a hierarchical definition of provided objects. The “dot” notation is used to separate different hierarchical levels.
- MultiZones. This approach is addressed to provide data sharing between different zones, for instance, in USA, there could be data sharing among school districts or between school districts and the state.

8.2.3.3.3 Security considerations

SIF has addressed security issues. The main limitation is that authentication support is optional, though the specification highly recommends it. The specification suggests using the underlying authentication procedures provided by the network (if applicable), in order to perform authentication quickly. If this is not the case, only the procedures associated to HTTPS are required by the specification.

Thus, security mechanism should be carefully defined in an environment which is to use the SIF approach, as the current specification leaves part of this issue open to implementers.

8.2.4 Suitability for Europe

8.2.4.1 Issues detected that are geographically dependent

Concerning SIF Architecture, no geographically dependent issues have been detected in the specification, as this part of the SIF specification deals only with the *zone model* for applications co-operating in an education environment, and its requirements in terms of allowed interactions. The SIF specification part where a substantial amount of geographically dependent issues may arise is the SIF data model, which is outside the scope of this report.

8.2.4.2 Issues in which a European standard should include multilingual support

In Europe, the way multilinguality is supported is very important. In relationship to architectural issues, SIF provides a means for sharing data based on XML messages. According to XML specification 1.0, “legal characters are tab, carriage return, line feed, and the legal characters of Unicode and ISO/IEC 10646”. Though SIF messages’ DTD does not specify any particular encoding for messages, the required HTTPS transport must include an HTTP header “content-type” specifying application/xml;charset= "utf-8", that is, UTF-8 encoding. In addition, SIF data objects’ DTDs do specify UTF-8 encoding. UTF-8 is an encoding designed for efficient transmission of the ASCII subset of Unicode: ASCII characters are encoded as a single byte; Latin characters with accents, Cyrillic, Greek, Hebrew and Arabic characters are encoded as two bytes and all other characters are encoded as three or four bytes. Thus, message contents are encoded with a character set that supports European languages.

But no explicit indication about the language that appears in textual elements of SIF messages is provided, nor is architecturally supported. If SIF environments are to be deployed, and multiple languages are involved, a solution must be provided for this. Several mechanisms are possible:

- Usage of “content_language” HTTP(S) headers.
- Extension of SIF messages and data definitions to include a “language” field or attribute.
- Include architectural support in SIF to handle multilinguality.

In addition to multilinguality issues, the current SIF Architecture shows several limitations (see before) in terms of how zones may be organized, and which applications may behave as data object providers in each zone. As it has been indicated, the current specification does not provide suitable support for all possible cases in the USA, so the application of SIF to European education institutions could arise similar concerns. Future SIF versions are targeted to solve this problem.

8.2.4.3 Issues that should be left out of a European standard

No issues have been found.

8.2.4.4 Issues that are in conflict, and Europe should set a different standard, or negotiate with the SIF a generalisation

See section 8.2.4.2 on multilingual support.

8.2.5 Compatibility with European developments

In this section several European developments are examined in order to check the compatibility and/or suitability of the SIF with them. A final section identifying complementary and in conflict issues is provided.

8.2.5.1 OASIS

Within OASIS, several areas have been selected as development examples of the project objectives: Open Code Software Library, Zone Management Server, School Server and Java-based kernel for supporting applications. In this section the suitability of the SIF architecture for usage in these areas will be examined.

8.2.5.1.1 Open Code Software Library

Within the OASIS developments, the Open Code Software Library (OCSL) will provide a web site offering information, material and support for school developers, teachers and users/providers interaction. The web site will include:

- Free and shareware Java based components to be used as components for the development of educational applications.
- Co-operative working tools to support developers, teachers and users/providers interaction.
- Information about commercially available tools and materials.
- Support for publication and/or acquisition of educational materials and developments.
- A “corner” for small educational Java applets made by teachers, groups promoted by Ministries and university groups (Teacher’s corner).
- A Distribution Service to distribute to the school community (broadcast, multicast, unicast): acknowledge, application, tools and training courses and materials, that will additionally allow to use it for project dissemination activities.

Several functions have been identified for the provision of the OCSL site:

- Acquisition and storage. Both automatic and manual acquisition of materials have been envisaged. Automatic acquisition involves the usage of data mining and harvesting techniques. Manual acquisition involves a help desk.
- Cataloguing. The EUN metadata modelling, based on XML, will be taken as a basis for material cataloguing.
- Dissemination. Pull and push technologies will be used. E-commerce support should be supported to allow applet providers the commercialisation of their products.
- CSCW support. Chats, news, mailing lists, conferences, help desks, ...

Due to the nature of OCSL purposes and functions, and considering the scope of application (potentially, European academic institutions and administrations), it seems reasonable to envisage that there will be a number of OCSL sites in Europe, which of course should communicate in order to share resources and materials. Each site will consist of several subsystems [7]: CSCW, Distribution, Teacher’s corner and Professional Developers.

With this framework, the suitability of SIF architecture has to be considered. SIF defines a framework for data sharing among applications. The following areas could benefit from a data sharing mechanism:

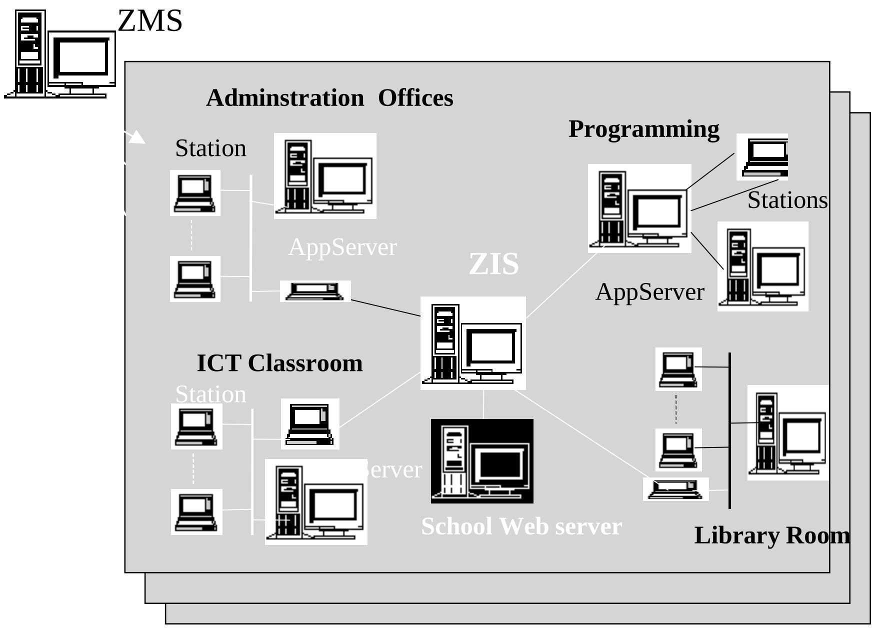
- OCSL web sites are intended to be used by members of the educational community. Thus, some site access control has to be provided. Information about people belonging to this community (e.g. teachers) would typically be kept in schools or in the relevant public institutions with educational competences (ministries ...). Information about people, thus, could be shared between these institutions and OCSL for identification of users. But, as it has been indicated before, the current SIF specification does not allow for hierarchical organization, so the application of SIF architecture to this model would require a single ZIS between OCSL and all personal information providers.
- Usage of SIF architecture and infrastructure for data sharing among OCSL sites and/or subsystems may be suitable, provided that SIF data model is extended to support the required data objects, and that multilingual support is somehow present. Scalability concerns may arise if the number of OCSL sites is high.
- Usage of SIF architecture for support of data harvesting application seems not applicable. The approach would work if SIF could provide data sharing associated to materials of interest to OCSL. This would require changes to SIF data model (but not architecture). But, in addition to that, it would only be applicable to data harvesting within nodes participant in a zone. If SIF architecture were extended beyond the current single-zone model, it could be used for data harvesting within the educational networks. Anyway, considering the objectives of OCSL, alternative data harvesting mechanisms should be included anyway, for data harvesting outside SIF zones associated to educational networks.
- Dissemination is restricted to members of the school communities accessing OCSL, thus they have to belong to the SIF environment. Depending on how OCSL deployment is organized, the usage of SIF architecture to support dissemination may show scalability problems due to the lack (at current specification state) for hierarchies.
- Suitability of the SIF architecture for support of the Teacher’s corner would focus mainly on information about updates in contents of the corner (educational applets). This would require that the SIF data model

is extended to support these data objects, and that multilingual support is somehow present. Scalability concerns may arise if the number of OCSL sites is high.

- In OCSL it is envisaged that there will be interaction with EUN metadata networking mechanisms. See section 8.2.4.2 for details on these mechanisms and SIF.

8.2.5.1.2 Zone Management Server

OASIS models the Networked School Community as a distributed networking system. Within this framework the network management and administration system is a distributed networking system, that consists of a Zone Management Server (ZMS) and one or more components organised into a zone. The size of the zone is flexible and could consist of a single building, school, a small group of schools, a district, etc. It is a scalable solution and supports multiple ZMS implementations or zones with ZMS interconnections. The components are the hardware and software components of the managed schools (PCs, servers, routers, operating systems, software tools and applications, etc.).



- SIF data sharing mechanisms allow request-response mechanisms for access to data objects.
- SIF data sharing mechanisms allow event reporting, associated to data updates.

These interactions are typical of a management environment, so the architecture developed by SIF could be used to support management. But, there are several drawbacks for this solution:

- SIF interactions do not support the concept of *asking* for data changes: they provide a means for notification of data updates, and for requesting data, but **not** for requesting data changes (i.e. the "SET" management operation). This excludes many management operations, and limits the applicability of this approach to monitoring.
- SIF data model would have to be extended for supporting most managed objects. This is not a limitation of the architecture, rather of the current data model.
- The current SIF architecture does not support hierarchical organizations, and this could be a limitation in large-sized management environments.
- Security considerations (see 8.2.3.3.3) need to be taken into account.

8.2.5.1.2.3 Collaboration between ZMS and SIF zones

Another possible approach is the collaboration between ZMS and SIF architecture, in such a way that SIF can be used for support of ZMS. This approach could be implemented as follows: in a SIF zone there could be an agent that would interact with other SIF agents in order to collect specific information of interest for ZMS, and then would interact via management protocols with the ZMS to provide the collected results, as shown in the figure below:

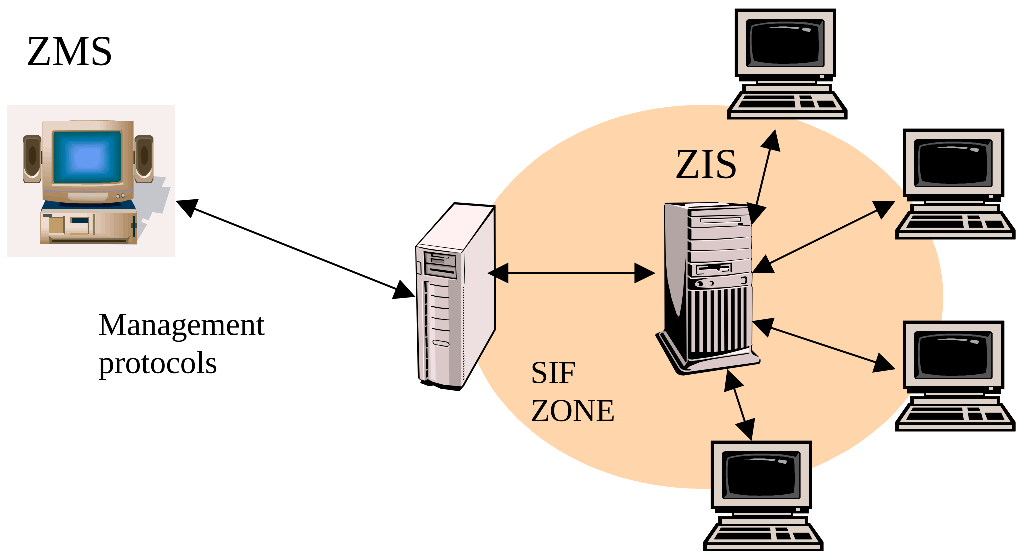


Figure 9: Collaboration between ZMS and SIF zones.

This approach seems promising, but there is still the issue of SIF interactions not allowing for "SET" operations.

Any of the SIF-ZMS cooperating approaches would require, in addition, the definition of a new working group within SIF, which should address computer and communication infrastructures. SIF data objects defined by this group could be monitored by ZMS using the SIF-based approach or the SIF-ZMS collaboration approach.

8.2.5.1.3 School Server

The SIF architecture, in its current state, could be suitable for implementation in school servers considering that, for this case, a SIF zone would roughly coincide. A possible approach is:

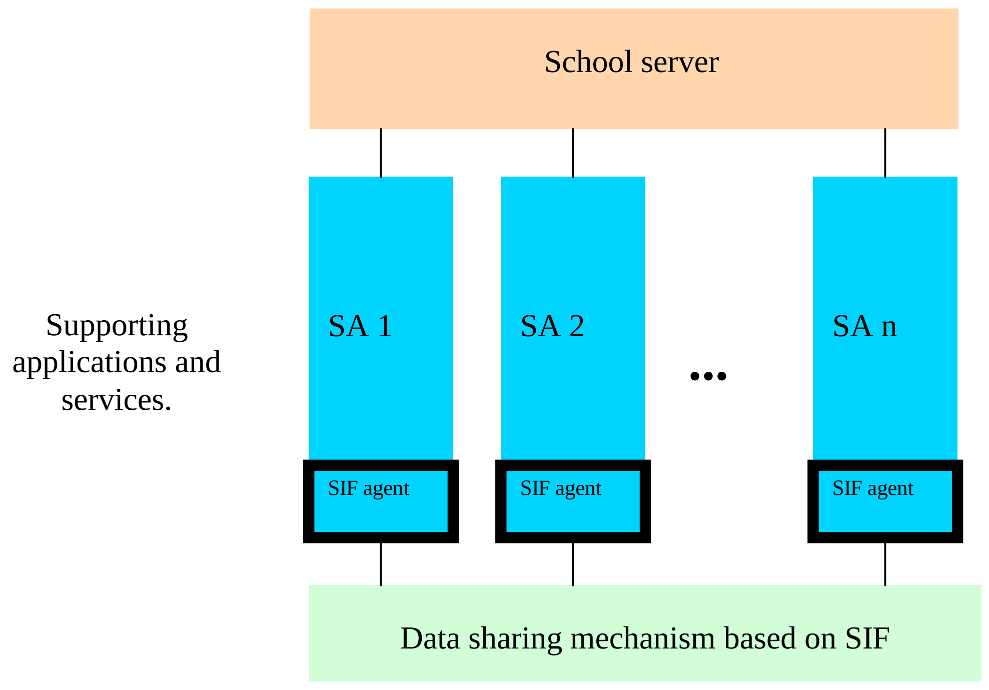

School server
Supporting applications and services.
SA 1
SA 2
...
SA n
SIF agent
SIF agent
SIF agent
Data sharing mechanism based on SIF

Figure 10: School server architecture based on SIF.

This approach shows the school server interacting with the supporting application and services. These applications and services will handle data, that may be shared among them using the SIF approach. This solution is conditioned to the suitability of SIF data model for European scenarios, including multilinguality. Another issue to be considered is the availability of software components supporting SIF agents.

A different approach would not impose that supporting elements of the school server would have all to be SIF-compliant. This approach would require a “bridge” between the SIF environment and the non-SIF environment, as shown in the figure below:

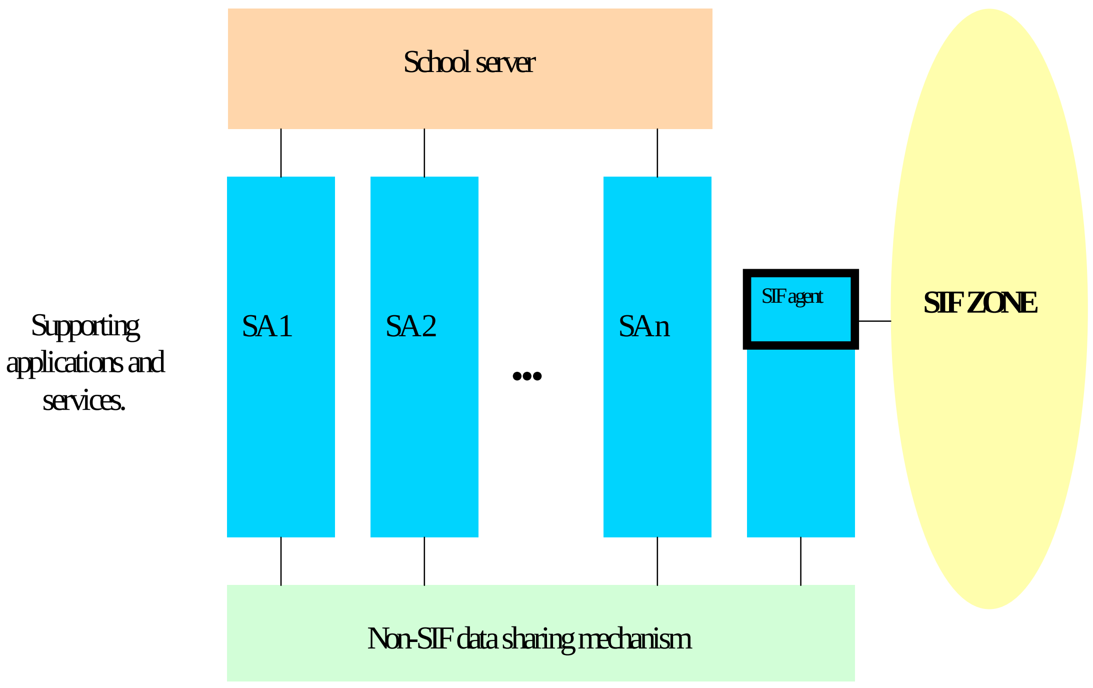

School server
Supporting applications and services.
SA1
SA2
•••
SAn
SIF agent
SIF ZONE
Non-SIF data sharing mechanism

Figure 11: School server architecture with SIF collaboration.

This approach does not condition the building and set-up of the school server to the availability of SIF-compliant supporting applications, but it does not exclude a SIF environment in the school, as a SIF-based application could be a component of the school server, communicating with the other components according to the implemented platform procedures, thus providing interaction between school server (and its associated applications and services) and the SIF-compliant applications.

8.2.5.1.4 Java-based kernel for supporting applications

This example of OASIS component focuses on the Virtual Workspace Environment (VWE). The VWE is an environment for flexible learning and collaboration. The VWE environment is developed in order to be used with the web browser and it is developed using Java, CORBA and XML. The development relies on the assumption that the teacher must be able to handle his/her own working environment. This is accomplished by the development of an environment (VWE) where the teacher constructs a Workspace by using a toolbox filled with components. VWE consists of two parts, the Kernel (maybe compared to a Operating System for the network) and Tools (may be compared to the software on a desktop computer.

The VWE is a platform independent environment, written in Java and CORBA. The VWE is built by distributed software components that provide the functionality. This approach is chosen in order to attain maximum flexibility in terms of functionality, adaptation and duration for the future.

One of the objectives of the Virtual Workspace Environment is to move the use of ICT in education from the desktop to the network. The purpose of this is to enhance administration, collaboration and flexibility.

VWE relies on CORBA and Java RMI for communication and cooperation between involved entities. SIF may have a complementary role in the deployment of VWE as, with the addition of a SIF-compliant agent, VWE could interact and share data with other SIF-compliant applications, thus allowing VWE users to access data from SIF applications.

8.2.5.2 EUN

The European SchoolNet (EUN) initiative [9] is targeted to build the EUN network of European national school networks. It should offer solutions for the regional, national and European trans-national school networks on progress towards a seamless Schools Learning Space, where different School Learning Communities can engage into knowledge building and networking projects.

Reference [3] outlines the technical aspects regarding the establishment of the EUN platform: overall architecture and some of the supported services. The EUN technical platform consists of a number of servers communicating through a network. The servers support several resources: information, tools, applications and services. Technical work in EUN platform aims at creating a logical structure and ensure the collaboration of national content/service providers to establish an entry to and a coherent view of a multitude of services and information.

The basis of all communication between the user and the EUN platform is the Internet based on the TCP/IP stack. The EUN resource network is driven by a combination of national initiatives and of central organization. It is based on the idea that new content and services will grow primarily from the initiatives and needs of the users. New services and content are initiated locally, and information on its existence propagates upwards to (or is harvested by) the national and official servers, thus becoming part of the resources of the EUN platform.

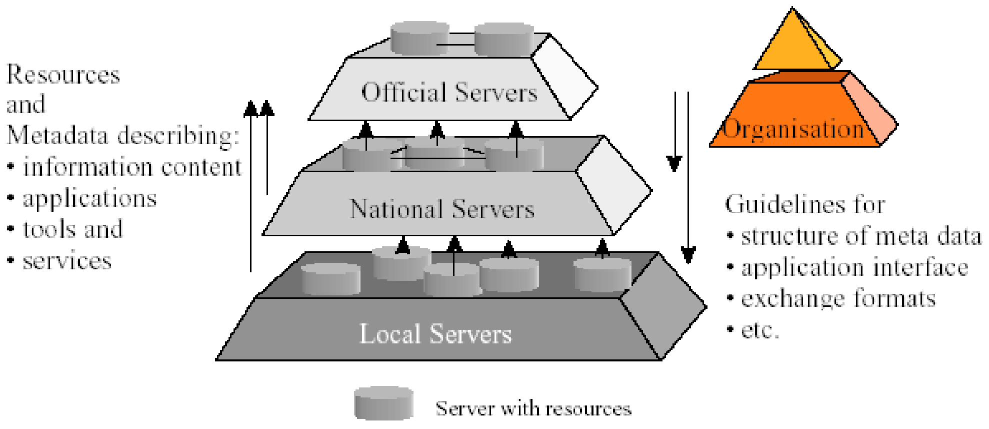


Figure 12: EUN as a resource network.

Multilinguality, authentication and access control are identified in [3] as issues that must be addressed in EUN.

The degree of SIF suitability for EUN network purposes has to be examined. SIF architecture provides a means for sharing data between applications. SIF-compliant applications may behave as providers of data, and other interested SIF applications may subscribe to events related to these data, and/or request data to the responder applications. Concerning only architectural issues, this mechanism could be suitable for usage in some parts of the EUN network (for instance, communication between local and national servers). But the limitations of current SIF specification should be taken into account: coherence has to be carefully handled, scalability issues may arise, and proper support of multilinguality may be needed. Of course, SIF data model should be extended to handle the resources and metadata interchanged

One of the areas addressed in EUN is the metadata networking infrastructure of the European Treasury Browser (ETB) project [10]. This infrastructure will support the interchange of metadata between a number of quality information providers, and it will provide a mechanism for supporting the submission of material to be approved by ETB. “Metadata” is structured descriptive information about a resource, and here these resources are educational resources and materials. The exchange of educational resource metadata between existing repositories throughout Europe is the key to the success of the European Treasury Browser project. In the framework of the ETB project, the metadata distribution and synchronization mechanism is based on NNTP. The use of NNTP in ETB is intended as a transfer protocol for XML metadata messages. As shown in figure 13, national node behave as clients of a core NNTP network. The core NNTP network consists of several interconnected NNTP servers, thus providing increased robustness.

SIF architecture is similar in objectives to this approach, as it provides a means for interchanging XML messages. But at present the model relies on a single ZIS per zone, so the SIF approach would be limited to SIF zones associated to national nodes. Trying to support the core network using a SIF approach implies a single ZIS, which should support all the load offered by the national nodes, thus becoming a bottleneck. Also, SIF data model would have to be extended to support the metadata objects handled by ETB. Another matter of concern is if the interactions in the ETB metadata network fit those defined in SIF.

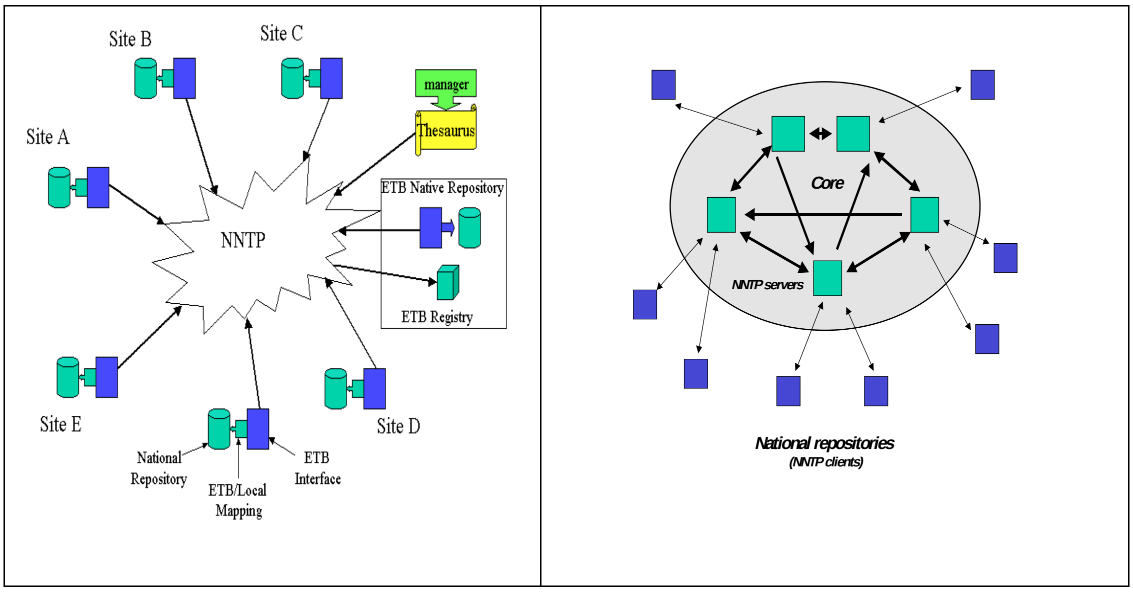


Figure 13: Architecture of ETB metadata network.

8.2.5.3 Complementary and in conflict issues

According to the review done up to this point, SIF provides an infrastructure mechanism for data sharing between applications. As far as architecture is concerned, the current SIF specification could be extended in terms of scalability (e.g. based on hierarchical organization) and interaction model (explicit “set” requests), in such a way that several European developments could benefit more from it. Other issues, like multilinguality and data model, have to be taken into account if SIF is to be adopted in Europe.

8.2.6 Recommendations

The conclusions of this report are:

- SIF architecture offers a way for school applications to share information in a robust, secure and efficient fashion.
- Concerning SIF architecture, the current specification is suitable for small scale deployments, but ongoing work is being done to provide hierarchy support and multizone communication to SIF zones. These developments should be tracked as they may be important for some OASIS scenarios.
- The current communication mechanisms allow data transfer encoded in UTF-8, so contents written in European languages can be encoded over SIF messages. But no multilinguality support is provided by the architecture itself. Even the identification of the language in which data carried in SIF messages is written is not provided by the current specification.
- Suitability of SIF architecture for support of management applications is restricted to monitoring, as SIF interactions do not allow to request data modification. If control is required, “Request” interactions defined in SIF should be extended to handle the concept of the “SET” management operation.
- Data object definitions would have to be extended in some OASIS example developments (e.g. if ZMS-SIF collaboration scenarios are supported). New SIF working groups can address this issue.

8.2.7 Other comments

8.3 Appendix C: SIF Infrastructure by OASIS

8.3.1 Introduction

8.3.1.1 Scope

The scope of this document is Work Package 2000 of the OASIS project.

8.3.1.2 Purpose

The purpose of this document is to provide a report on the SIF Infrastructure and its relevance to the OASIS project. The document is organized as follows: first, a summary of infrastructure issues of the SIF specification is provided; then, the relationship between SIF and European educational developments and initiatives (OASIS among them) is reviewed; finally, the conclusions are presented.

8.3.1.3 Glossary

• CORBA	Common Object Request Broker Architecture,
• CSCW	Computer Supported Cooperative Work.
• EU	European Union.
• EUN	EUropean schoolNet.
• HTTP	HyperText Transfer Protocol.
• HTTPS	HyperText Transfer Protocol, Secure.
• OASIS	Open Architecture and Schools In Society.
• OCSL	Open Code Software Library.
• PC	Personal Computer.
• RMI	Remote Method Invocation.
• SIF	Schools Interoperability Framework.
• SMTP	Simple Mail Transfer Protocol.
• USA	United States of America.
• VWE	Virtual Workspace Environment.
• XML	eXpress Markup Language.
• XSLT	eXtensible Stylesheet Language Transformations.
• ZIS	Zone Integration Server.
• ZMS	Zone Management Server.

8.3.2 Sources

1. Schools Interoperability Framework. Implementation Specification. Version 1.0 Revision 1. August, 2001. http://www.sifinfo.org
2. SIF Standard for K-12. Where do we go from here? Ron Kleinman, 21/1/2002.
3. OASIS Shared Cost RTD Project. Annex 1. Description of Work. 19/7/2001.
4. Extensible Markup Language (XML) 1.0 (Second Edition) W3C Recommendation, 6th October 2000. http://www.w3.org/TR/REC-xml
5. SIF Architecture Report. OASIS WP2000 work document.
6. EUN Multimedia MM1010. Deliverable: EUN Architecture and Framework D06.02.01. May, 2000.
7. European Treasury Browser. Deliverable D9.1: WP9 Architecture of the Metadata Networking Infrastructure. October 2000.

8.3.3 Procedure in analysis

In this section the SIF Infrastructure is reviewed in order to identify its main characteristics and the relevant issues for the OASIS project.

8.3.3.1 Introduction to SIF

Schools Interoperability Framework (SIF) [3] is the place where companies and educators will be able to participate in the development of specifications that will ensure interoperability in the K-12 instructional and administrative environment. The aim of a SIF implementation is to provide applications related to schools or education with effective means of sharing data. In the USA, where this specification has been developed, some example areas of these applications are: student information systems, food services, transportation services, library applications, etc.

8.3.3.2 SIF infrastructure

According to SIF specification, Infrastructure develops and specifies a standard framework of messages and communication mechanisms that allow diverse applications for the education industry to effectively, reliably, and securely exchange information over open networks in a platform- neutral manner.

Section 4 of the SIF specification is devoted to infrastructure details. SIF infrastructure consists of:

- Common elements.
- Messages.
- Objects.

8.3.3.2.1 Elements

There are two elements that are common to all SIF messages:

- SIF_Header. Common message header for all SIF messages. It carries information related to message identification, date and time associated to the message, source identification, destination identification, and security information.
- SIF_Message. Outer most tag in all SIF messages. It has an attribute specifying the XML [4] namespace for SIF messages.

8.3.3.2.2 Messages

These messages are defined in SIF:

- SIF_Ack. Acknowledgement for infrastructure messages. All infrastructure messages will return a SIF_Ack as a result to indicate if the request was successful or not. If successful, the SIF_Ack can include additional information to be sent to the caller. If an error is reported, the SIF_Ack will include one or more SIF_Error elements.
- SIF_Event. It is used to deliver event (add, change, delete) objects. It includes the actual objects (partial or whole) that are sent along with the SIF_Event.
- SIF_Provide. It is used for advertising the provision of data objects.
- SIF_Register. It is used for agent registration with a ZIS. It includes identification information and all relevant information for the ZIS to contact the agent: SIF version, buffer size, mode (push/pull), protocol information.
- SIF_Request. It is used to request information in SIF data objects from other agents. It specifies fields in the object to request and the query or search criteria. Support for specification of conditions to be satisfied by the requested objects is provided: complex queries are built by using AND and OR Boolean operators. Each simple condition is tested by using an operator which, in the current SIF specification (1.0r1) is restricted to equality (EQ). Other condition operator or queries may be included in future versions of the specification. Elements and attributes of an object are designated by using a pattern language based on the XSLT patterns syntax.
- SIF_Response. It is used to respond to a SIF_Request message. It consists of several data objects, which are complete or partial. SIF_Error elements may be present if any error conditions have occurred while processing the SIF_Request. A response may be divided into multiple SIF_Response messages only if the response is so long that sending it in a single message may cause performance problems or longer latency.
- SIF_Subscribe. It is used to indicate subscription to a specified object.
- SIF_SystemControl. It is used to control the flow of data between two SIF nodes. In fact, it is a container for specialized control submessages. These include: SIF_Ping, to detect if an agent or ZIS

is ready to receive and process messages; SIF_Sleep, used to signal the receiver that it must not send any more messages to the SIF_Sleep sender; SIF_Wakeup, used to signal that a sleeping agent or ZIS is ready to resume message processing; SIF_GetMessage, to provide an agent with a mechanism for pulling a message from a ZIS.

- SIF_Unprovide. It removes the message sender as a provider of the data object contained in the message.
- SIF_Unregister. It allows an agent to remove any association with a ZIS. The ZIS will remove all provisions and subscriptions it maintains for the sender.
- SIF_Unsubscribe. It removes the message sender as a subscriber to the SIF_Events contained in the message.

8.3.3.2.3 Objects

There is only one object defined for SIF infrastructure:

- SIF_ZoneStatus. It is an object implicitly provided by all Zone Integration Servers to provide information about the ZIS. It includes such information as ZIS vendor name, product and version, zone name and identifier, and information about the zone status:
    - Registered providers and the objects provided by each provider.
    - Registered subscribers and the event objects subscribed by each subscriber.
    - All nodes attached to the ZIS: name, identifier, type, communication mode (push/pull), communication protocol and associated security, message processing status (sleeping), maintained statistics (optional, and with no defined child elements in version 1.0).
    - ZIS supported protocols, security mechanisms, and SIF versions.

That is, this SIF infrastructure object can be envisaged as a container for zone/ZIS monitoring information.

8.3.3.3 Comments

It is worth repeating here that SIF infrastructure aims at providing the diverse applications for the education industry to effectively, reliably, and securely exchange information over open networks in a platform-neutral manner. This is achieved by the definition of request/response mechanisms, as well as information update mechanisms based on publishing/subscription. SIF Infrastructure provides messages for these data exchange mechanisms. SIF messages are based on XML, which is an open standard, without any ties to an specific platform. SIF XML messages can be carried over a variety of transport protocols. HTTPS support is mandatory, but other protocols (e.g. HTTP, SMTP ...) could be used as transport protocols for SIF XML messages, and in fact several commercial SIF platforms support this (e.g. Microsoft SIF implementation allows the usage of SMTP as message transport). Anyway, the fact that HTTPS, which is an open standard, is mandatory, guarantees the provision of a secure information exchange mechanism over open networks, without being tied to a specific platform.

8.3.3.4 Limitations of SIF version 1.0r1

The main infrastructure-specific limitations of the SIF specification are related to the current state of the definition of messages and objects:

- Queries are restricted to a subset of logical conditions (at present, only equality is defined as a test mechanism).
- ZoneStatus maintained statistics for nodes at present are not expanded into any specific value.

Nevertheless, most applications can work without taking into account these limitations, so their impact can be considered as minor.

There is another possible limitation issue concerning size of SIF messages. There is a parameter, SIF_MaxBufferSize, used when an agent registers itself at the ZIS, indicating the maximum size of SIF messages that the agent is able to accept. The ZIS checks that this value is large enough for participation in the SIF zone. This parameter is also passed in other SIF messages (e.g. SIF_Request), and if so, it must be consistent with the value indicated in SIF_Register.

The specification also indicates that a SIF_Response may be divided into several messages if necessary, in order to minimize latency and maximize performance. Other messages, like SIF_Event, do not offer this possibility, that is, if their size exceeds SIF_MaxBufferSize for the event subscriber, there will be an error. Anyway, SIF suggests that only those elements of a data object that are really updated should be carried in the event message, in order to reduce as much as possible the total message size.

Other limitations are not directly related to infrastructure, but to architectural and data model issues, as the SIF infrastructure is simply the platform for provision of the defined interactions in the SIF infrastructure and the data objects defined by the data model.

8.3.4 Suitability for Europe

8.3.4.1 Issues detected that are geographically dependent

No specific geographical dependent issues have been detected in the infrastructure section of the SIF specification.

8.3.4.2 Issues in which a European standard should include multilingual support

As far as the Infrastructure section of the SIF document is concerned, no issues have been found. In general, SIF messages must be able to carry multilingual information. This must not be understood as imposing any redefinition on SIF message tag names, rather it imposes a requirement on the ability of SIF messages to carry information contents in languages other than English.

8.3.4.3 Issues that should be left out of a European standard

As far as the Infrastructure section is concerned, no issues have been found.

8.3.4.4 Issues that are in conflict, and Europe should set a different standard, or negotiate with the SIF a generalisation

As far as the Infrastructure section is concerned, no issues have been found.

8.3.5 Compatibility with European developments

In this section several European developments are examined in order to check the compatibility and/or suitability of the SIF with them. A final section identifying complementary and in conflict issues is provided.

8.3.5.1 OASIS

Compatibility issues between OASIS developments and SIF architecture have already been discussed in other document. Concerning infrastructure, no specific issues have been detected, as SIF infrastructure is directly conditioned by SIF architecture and data models. That is, compatibility issues between SIF infrastructure and OASIS are directly related to the definition of SIF architecture and data models. The next paragraphs deal with several developments within OASIS and the issues involving SIF infrastructure, continuing with the approach initiated in [5].

8.3.5.1.1 Open Code Software Library (OCSL).

As indicated in [5], suitability of SIF for supporting OCSL is mainly envisaged for data sharing between OCSL subsystems, and conditioned to the extension of the SIF data model for supporting the shared data. As far as SIF infrastructure is concerned, this implies that messages related to SIF objects (e.g. for publishing, requesting, responding, reporting events …) will carry information or contents of these shared data. There is no problem provided that these data are defined in the SIF data model. SIF infrastructure messages provide a means of carrying information or content of SIF data objects, and refer to them by their object names and attributes, and the specific individual object has its RefId (see [1]).

The interaction model defined by SIF infrastructure may also condition the application to the OASIS OCSL. For instance, applications related to download of a large-sized object, even if the SIF data model is updated to be able to describe it, may be conditioned by the limitations imposed by the SIF_MaxBufferSize parameter of SIF messages. This value can be indeed very high (some examples in the SIF specification hold values of about 1 Mbyte), but this would depend on the implementation. SIF infrastructure messages, thus, would be useful for handling interactions associated, for instance, to metadata, OCSL objects (software, applets, courses, ...) descriptors, etc., but not for direct transfer of the OCSL contents.

8.3.5.1.2 Zone Management Server (ZMS).

As indicated in [5], suitability of SIF for supporting ZMS could involve two approaches:

- SIF as architectural support for ZMS.
- Collaboration between SIF zones and ZMS.

The first approach deals with a SIF-based platform for interaction between ZMS and managed objects. The main drawback of this approach was the lack in SIF of a proper remote control model. Update events could provide an indirect way of doing “SET” management operations, but the SIF specification does not provide a explicit SET, and events could arise a potential consistency problem, as SIF specification does not enforce data consistency [2]. Support for remote execution of commands is not directly provided in SIF, either, and it would have to be emulated by using similar mechanisms as those in SNMP platforms (e.g. by doing a SET in SNMP or its SIF equivalent based on events). Anyway, if this approach is selected, it has the same limitation as before: usage of SIF infrastructure requires that involved data be defined in SIF data model.

The second approach deals with a ZMS interacting (by standard management protocols) with a managed object which in turn results to be a SIF agent, thus being able to participate in a SIF zone. This agent would be a kind of proxy between the SIF zone and the ZMS. Anyway, every management operation that this proxy could initiate, should be based in SIF infrastructure messages on the SIF zone, and thus, it must refer to data objects defined by the SIF data model.

Common to both approaches, there are several issues to take notice of:

- For SIF monitoring, SIF infrastructure does provide an object, SIF_ZoneStatus, provided by the ZIS, that contains information about current zone status. Information contained in this object must be provided somehow to the ZMS. See section 8.3.3.3 for limitations of the current definition of this SIF_ZoneStatus object.
- If SIF data model is extended for supporting new data objects related to management, a concern may arise related to if any kind of new object could be supported by the SIF infrastructure or if there are restrictions in the definition. As far as the object is present in the SIF data model, the only restriction that would apply to data sharing (apart from access control) is that the SIF messages must not be larger than the SIF_MaxBufferSize parameter indicated by an agent. The ZIS has also the possibility of assuring that the SIF_MaxBufferSize that an agent tries to register is large enough for participation in the SIF zone. See 8.3.3.3 for more information.

8.3.5.1.3 School Server.

The SIF architecture, in its current state, could be suitable for implementation in school servers considering that, for this case, a SIF zone would roughly coincide. As stated in [5] the possible approaches are:

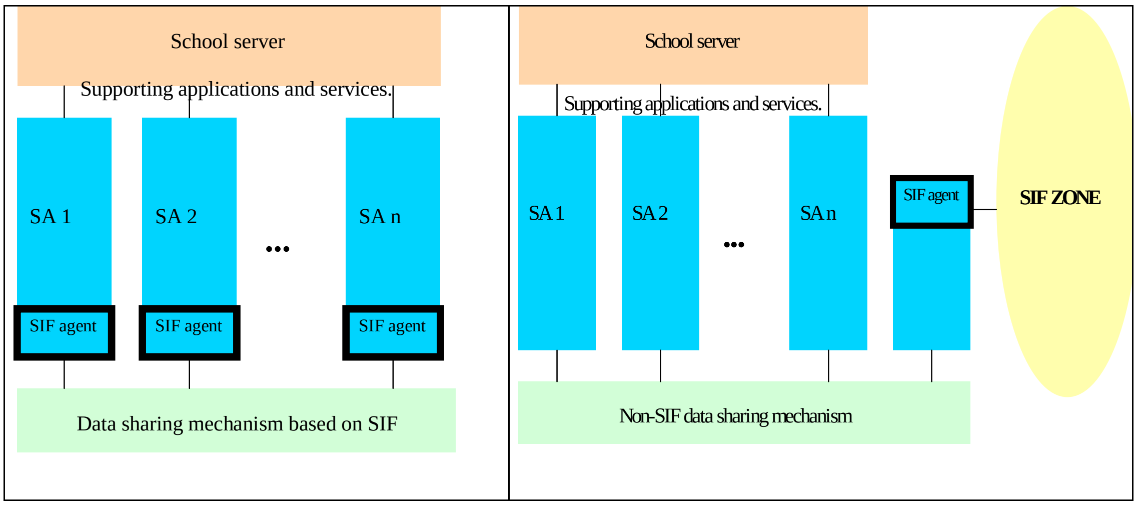

School server
Supporting applications and services.
SA 1
SA 2
SA n
...
SIF agent
SIF agent
SIF agent
Data sharing mechanism based on SIF
School server
Supporting applications and services.
SA 1
SA 2
SA n
...
SIF agent
SIF ZONE
Non-SIF data sharing mechanism

Figure 14: School server architecture based on SIF (left) and with SIF collaboration (right)

These approaches show the school server interacting with the supporting application and services. These applications and services will handle data that may be shared among them using the SIF approach. Both solutions are conditioned to the suitability of SIF data model for European scenarios, including multilinguality. Thus, the compatibility between SIF and the School Server requires that shared data be defined in the SIF data model, so that SIF infrastructure messages can carry information on them.

As before, SIF_MaxBufferSize also imposes constraints on what can be directly transferred with SIF interactions.

8.3.5.1.4 Java-based kernel for supporting applications

This example of OASIS component focuses on the Virtual Workspace Environment (VWE). The VWE is an environment for flexible learning and collaboration. The VWE environment is developed in order to be used with the web browser and it is developed using Java, CORBA and XML. The development relies on the assumption that the teacher must be able to handle his/her own working environment. This is accomplished by the development of an environment (VWE) where the teacher constructs a Workspace by using a toolbox filled with components. VWE consists of two parts, the Kernel (maybe compared to a Operating System for the network) and Tools (may be compared to the software on a desktop computer.

The VWE is a platform independent environment, written in Java and CORBA. The VWE is built by distributed software components that provide the functionality. This approach is chosen in order to attain maximum flexibility in terms of functionality, adaptation and duration for the future.

Notice that Java and CORBA have their specific remote method invocation and remote procedure call mechanisms and libraries. The approach outlined in [5] is to SIF may have a complementary role in the deployment of VWE as, with the addition of a SIF-compliant agent, VWE could interact and share data with other SIF-compliant applications, thus allowing VWE users to access data from SIF applications. As far as infrastructure is concerned, this approach is the most suitable, as SIF infrastructure does not provide messages for provision of remote procedure call or method invocation.

8.3.5.2 EUN

The European SchoolNet (EUN) initiative [9] is targeted to build the EUN network of European national school networks. It should offer solutions for the regional, national and European trans-national school networks on progress towards a seamless Schools Learning Space, where different School Learning Communities can engage into knowledge building and networking projects.

Reference [3] outlines the technical aspects regarding the establishment of the EUN platform: overall architecture and some of the supported services. The EUN technical platform consists of a number of servers communicating through a network. The servers support several resources: information, tools, applications and services. Technical work in EUN platform aims at creating a logical structure and ensures the collaboration of national content/service providers to establish an entry to and a coherent view of a multitude of services and information.

In reference [5] several SIF architecture limitations were examined when considering suitability for implementation of the EUN network.

One of the areas addressed in EUN is the metadata networking infrastructure of the European Treasury Browser (ETB) project [10]. This infrastructure will support the interchange of metadata between a number of quality information providers, and it will provide a mechanism for supporting the submission of material to be approved by ETB. In the framework of the ETB project, the metadata distribution and synchronization mechanism is based on NNTP. The use of NNTP in ETB is intended as a transfer protocol for XML metadata messages.

SIF is similar in objectives to this approach, as it provides a means for interchanging XML messages. As indicated in [5], at present the model relies on a single ZIS per zone, so the SIF approach would be limited to SIF zones associated to national nodes. Trying to support the core network using a SIF approach implies a single ZIS, which should support all the load offered by the national nodes, thus becoming a bottleneck. Also, SIF data model would have to be extended to support the metadata objects handled by ETB.

Nevertheless, SIF infrastructure messages suitability for the EUN metadata networking should be considered, as provides a mechanism for data sharing and updating of objects that are described in XML. SIF_MaxBufferSize parameter can be, again, an issue here, as potentially some large objects could be exchanged (e.g. according to [4], new thesaurus versions would be transferred as a single XML record). SIF infrastructure mechanisms for handling updates (e.g., reporting *only* the changed part of an object) could be of interest here.

8.3.5.3 Complementary and in conflict issues

According to the review done up to this point, SIF provides an infrastructure mechanism for data sharing between applications. As far as infrastructure is concerned, the current SIF specification issues that may have influence in some European developments concern SIF maximum message sizes, which is a parameter depending on the SIF implementation. But it should not be considered as a conflict, rather it is an issue that should be taken into account when considering SIF deployments in the specific environments of these European developments.

8.3.6 Recommendations

The conclusions of this report are:

- SIF Infrastructure provides the specification of messages supported by a SIF platform.
- As such, its features and limitations are directly conditioned by the definition of architecture and data model of the SIF platform. Specifically, all data objects that can be carried over SIF infrastructure messages have to be defined in the SIF data model.
- The current SIF specification (1.0r1) has some infrastructure messages that have to be further developed and refined in future versions. Nevertheless, the impact of these issues is minor.
- Implementation considerations regarding message size could be an issue for usage of the SIF infrastructure.
- No geographical dependent issues have been detected, as far as infrastructure is concerned.

8.3.7 Other comments

8.4 Appendix D: HTTPS and its use by the SIF Infrastructure transport layer by OASIS

8.4.1 Introduction

The Hypertext Transfer Protocol over Secure Socket Layer (HTTPS, or HTTP over SSL) is a web protocol originally developed by Netscape and built into web browsers that encrypts and decrypts user page requests as well as the pages that are returned by the web server. HTTPS is the use of Secure Socket Layer (SSL) for web connections via a sub-layer under a normal HTTP application layering. HTTPS uses port 443 instead of HTTP port 80 in its interactions with the lower layer at the TCP/IP level.

HTTPS and SSL support the use of X.509 digital certificates from the server so that, if necessary, a user can authenticate the sender. SSL is an open, non-proprietary protocol that has been proposed as a standard to the World Wide Consortium (W3C). HTTPS is not to be confused with S-HTTP, a security-enhanced version of HTTP developed and proposed as a standard by the Internet Engineering Task Force (IETF).

8.4.2 SIF Infrastructure and HTTPS

The SIF Infrastructure messages are used by SIF to encapsulate and transfer data objects. Together, they form a messaging application program interface (API), which is expressed in XML.

The design objective for SIF is to express the entire Infrastructure API in XML. SIF should not have dependencies upon any underlying transport layer that could provide functionality other than the transportation of the XML from client to server and back. This ensures that SIF Infrastructure messages can be carried over a variety of communication transports.

The SIF Infrastructure depends upon the transport layer to provide a reliable connection to move messages back and forth from client and server. The transport layer is also responsible for providing data security by means of data encryption and authentication of the client and server. Some transport layers even provide data compression, which is an important factor when processing a large volume of XML messages.

Delegating the authentication, compression, and encryption to the transport layer, makes the user interface to the transport simpler. A client that wishes to send an infrastructure message assembles the message in XML and then hands it off to the transport layer for delivery. The transport layer takes the XML message and transfers it to the server where it is taken from the transport layer and processed.

In moving from the client to the server, the transport may have compressed, encrypted, and authenticated the connections but all of this is transparent to the users of the SIF Infrastructure API. To the user, it is XML in and XML out.

Different types of transports are or will become available providing various features and benefits. An Agent or Zone Integration Server (ZIS) may employ multiple transport protocols but they must support SIF HTTPS.

8.4.3 The SIF HTTPS Transport

In order to ensure that Agents and Zone Integration Servers can communicate with each other regardless of vendor or platform, all Agent and ZIS implementations must support the SIF HTTPS transport layer protocol.

SIF HTTPS combines the HTTP 1.1 protocol (RFC-2616) and the TLS 1.0 secure socket protocol (RFC-2246) resulting in an easy to use and secure transport protocol. Being based upon HTTP 1.1, the protocol supports persistent or keep-alive connections that greatly increase the message throughput between sender and receiver. This is an especially important factor when using TLS 1.0, which creates a significant amount of overhead during the initial opening of the connection.

When a SIF Agent is configured to use HTTPS, a request to use HTTP will result in another request for a secure channel. The HTTPS variation will be used in all situations where the client is talking to a web server and the web server requires HTTPS in order to ensure a secure connection.

Support of Transfer Encoding and data chunking (RFC-2616, Section 3.6) is not required for SIF HTTPS. An implementation of the protocol may support Transfer Encoding and data chunking, but it must be able to communicate successfully with a client or server that does not support this feature.

Because protocol changes are handled at the Infrastructure XML API level, a client or server must not use the "Connection: Upgrade" or "Upgrade: xxx" headers to invoke a request for a protocol change. If a client or server receives an upgrade header, it must ignore that header and not change communication protocols.

8.4.3.1 SIF HTTPS Request and Response Model

A client is defined as the party (Agent or ZIS) who initiates a connection to a remote machine. The remote end is known as the server. A client using the SIF HTTPS protocol opens a connection to the server and sends a HTTP 1.1 POST request with the SIF Infrastructure XML message as the POST payload. The server responds with a HTTP response with the Infrastructure XML acknowledgement message as the response payload.

The default behaviour for HTTP 1.1 is to use persistent or "keep-alive" connections. When operating in this mode, 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.

8.4.3.1.1 SIF HTTPS Request Headers

The following HTTP request and common headers defined in RFC-2616 must be present in all SIF HTTPS messages sent by a client:

HTTPS Request Headers

Header	Description	Recorred Contents
Content-Length	The exact size of the attached payload (XML message)	
Content-Type	Describes the contents of the request. Firewall and web server programs can filter messages going through a network by examining this header.	Application/xml;charset= "utf-8"
Host	Specifies the Internet host and port number of the destination server	

Table 5: HTTPS Request Headers

In addition to the above headers, a client may include a "Connection: close" header in the HTTP request if it wishes to close the current connection after receiving the response. If this header is included, the client must not send additional requests on this connection. The client should close the connection after receiving the response.

The client may include additional headers however the server must be able to successfully process POST requests that only contain the required headers.

SIF HTTPS Request

```
POST /MyPath HTTP/1.1
Content-Length: 467
```

```
Content-Type: application/xml;charset="utf-8"
Host: sifinfo.org:8000
<SIF_Message xmlns="http://www.sifinfo.org/v1.0r1/messages">
    <SIF_SystemControl>
        <SIF_Header>
            <SIF_MsgId>56409F0C01FBD1C44300B4518E100765</SIF_MsgId>
            <SIF_Date>20010411</SIF_Date>
            <SIF_Time Zone="UTC-05:00">18:18:13</SIF_Time>
            <SIF_SourceId>SifInfo_TestAgent</SIF_SourceId>
        </SIF_Header>
        <SIF_SystemControlData>
            <SIF_Ping/>
        </SIF_SystemControlData>
    </SIF_SystemControl>
</SIF_Message>
```

Implementations of this SIF HTTPS must be able to specify the value for the path ("/MyPath" in the example) as the Agent or ZIS may require a specific value for routing purposes.

8.4.3.1.2 SIF HTTPS Response Headers

The following HTTP response and common headers defined in RFC-2616 must be present in all SIF HTTPS responses messages sent by a server:

HTTPS Response Headers

Header	Description	Required Contents
Content-Length	The exact size of the attached payload (XML message)	
Content-Type	Describes the contents of the request. Firewall and web server programs can filter messages going through a network by examining this header.	Application/xml;charset= "utf-8"
Date	The current date and time in the format described in RFC-2616 Section 3.3. Note that the date is UTC based and NOT local time.	
Server	Identifies the server sending the response. Clients may use this information to infer information about the server being contacted (vendor, model, version, capabilities, etc.)	

Table 6: HTTPS Response Headers

In addition to the above headers, a server may include a "Connection: close" header in the HTTP response if it wishes to close the current connection after sending the response. The server should close the connection after sending the response.

The server may include additional headers however the client must be able to successfully process Response notices that only contain the required headers.

SIF HTTPS Response

```
HTTP/1.1 200 OK
Content-Length: 585
Content-Type: application/xml;charset="utf-8"
Date: 02 Apr 2001, 23:32:00 UTC
Host: sifinfo.org:4000
Server: SIFZIS;V1.0r1
<SIF_Message xmlns="http://www.sifinfo.org/v1.0r1/messages">
    <SIF_Ack>
        <SIF_Header>
            <SIF_MsgId>4A900E10F4E675CF4A01B4518E100765</SIF_MsgId>
            <SIF_Date>20010411</SIF_Date>
            <SIF_Time Zone="UTC-05:00">18:18:13</SIF_Time>
            <SIF_SourceId>SifInfo_TestZIS</SIF_SourceId>
        </SIF_Header>
        <SIF_OriginalSourceId>SifInfo_TestAgent</SIF_OriginalSourceId>
        <SIF_OriginalMsgId>56409F0C01FBD1C44300B4518E100765</SIF_OriginalMsgId>
        <SIF_Status>
            <SIF_Code>0</SIF_Code>
        </SIF_Status>
    </SIF_Ack>
</SIF_Message>
```

Although the SIF HTTPS protocol uses the 200-OK response notice to communicate all responses, Agent or ZIS implementations could be built using existing web server infrastructures. Therefore, SIF HTTPS implementations should expect to receive other HTTP 1.1 response notices.

8.4.3.1.3 Other Type of Response Notices

• The 100-Continue Notice:

> This response notice is generally sent if the client has included an "Expect: xxx" request header. A SIF HTTPS client may include an "Expect: xxx" header but generally does not. Certain web server implementations send a 100-Continue notice even though the original request did not contain an "Expect: xxx" header. When a client receives a 100-Continue notice, it must discard that response and wait for a subsequent 200-OK notice.

• The 3xx, 4xx, 5xx Notices:

> A server should only return 200-OK response notices but may return other notices. Servers built using existing web server technology are more likely to return other types of response notices. If a client receives any 3xx, 4xx, or 5xx response notices, it must treat these responses as if a transport error has occurred.

8.4.4 SIF HTTP Transport

The SIF HTTP protocol is identical to the SIF HTTPS transport but does not include the TLS 1.0 security layer, which provides data encryption and authentication. An Agent or ZIS may implement the SIF HTTP transport but must implement the SIF HTTPS protocol. SIF Infrastructure, because of the sensitive data being exchanged by schools, only supports the HTTPS protocol.

8.4.5 Recommendations and application in OASIS

The recommendations of this report are:

- The SIF transport layer, based on HTTPS, provides the exchange of messages supported by a SIF platform.
- To assure a minimum level of secure data exchange, the SIF Specifications require the use of HTTPS with a recommendation of 128-bit encryption. This is a secure and reliable level of security and all applications within any SIF Zone should adopt this minimum level of encryption.
- If any application within any given 'SIF Zone' uses a transport method with a less secure level transport method than HTTPS, the ZIS should reject any request either to provide information or to subscribe to information.
- The SIF Infrastructure Working Group is already into discussion to expand the possibilities of accepting other data transports methodologies such as SOAP. The implementation of SOAP would be studied to ascertain its impact, viability within the specifications.
- Implementation considerations regarding future expansion of the transport layer methodologies used by SIF should be directed to the SIF Infrastructure Working Group for consideration.

Recommendations for application development and testing:

- Applications in development trying to become SIF-able could test all their functionality within a test system using HTTP as the transport method to clearly separate functionality and information delivery issues versus secure transport layer issues.
- Once an application has fixed all its issues regarding functionality and information delivery then HTTPS should be selected as the minimum transport protocol of choice