Access 4 Learning Community

Powered by: SIF

Simple. Secure. Scalable. Standard.

SIF Infrastructure Specification 3.3: Read This First

www.A4L.org

Version 3.3, May 2019

Introduction

This SIF 3.3 Infrastructure *Read This First* (RTF) document contains a set of Frequently Asked Questions (FAQs) and their answers. Together they are designed to assist a technical readership ranging from those only casually interested in the SIF standard to those actively planning to adopt it, in quickly "getting their arms around" a secure, featureful REST-based infrastructure that took four volumes of documentation to completely describe.

This RTF serves as a non-normative "quick start" guide, which does not replace the SIF specification. It reorganizes, summarizes, interprets and at times expands on the material contained in the other volumes without introducing new infrastructure requirements. As such, the RTF can be reissued as additional important questions get raised, without requiring the reissuing of a new release of the Infrastructure. It currently provides:

- An overview of the infrastructure architecture and a definition of common terminology
- A brief summary of the individual artifacts comprising the SIF 3.3 Infrastructure release (documents, schema and sandbox), and references to additional (supplementary) developer collateral.
- A set of "aspects" of the SIF 3.3 Infrastructure release (such as *Security* or *Scalability)*, which define how each aspect is supported in the SIF standard, and includes specific references to where it is addressed in the SIF 3.3 infrastructure documentation.

Changes in this Release

This release adds:

1. Privacy Integration
2. Version Indication/Negotiation
3. PESC JSON Adoption
4. Cleaner, More Consistent Schemas and Artifacts[1]
5. Clarifications[2]

Documents explaining the rational and impacts of these changes can be found towards the end of the Specification website here.

Read this First

For those REST developers new to the SIF 3 standard and interested in minimizing the material they need to understand before getting started developing actual SIF 3 compliant software (i.e. those not willing to read this entire *Read This First* first), the following sequence of steps is suggested.

- Read chapters 1 through 3 below (the fundamentals, the architecture and the overview of SIF 3 Infrastructure artifacts and the change history since SIF 3.0). From that point forward, everything else should make a great deal more sense.

- The "other resources" information in section 3.5 should also be very useful to a developer trying to make a basic SIF component operational.

Those developers already familiar with the SIF 3 standard should start with section 1 of the Base Architecture document, which explains the purpose of the release.

[1] *Not intended to change the meaning of the specification.*

[2] *Not intended to change the meaning of the specification.*

1. The fundamentals

The questions and answers in this section address the overall vision of the SIF 3.3 Infrastructure release. They should be the starting point for most readers of the associated documentation.

1.1. What exactly IS the *SIF 3.3 Infrastructure*?

SIF 3.3 is the latest release of an open standard infrastructure, which began nearly 20 years ago as the product neutral interface of an existing commercial message broker. As will be seen, this release brings SIF into the modern era by leveraging a REST based approach to data exchange. The key contribution the SIF 3.3 release makes is to define, coordinate and standardize the ways in which multiple RESTful clients can access a RESTful educational service securely, robustly, and in real time.

The SIF 3.0 infrastructure was the first infrastructure release to be completely separate from the data model defining the payloads it carries, which means it can be used to support many different data models in many different locales. For example while the SIF US data model is based on CEDS, this is not explicitly reflected in the SIF 3 Infrastructure documentation.

1.2. Why was a major release of the infrastructure needed?

The SIF 2.x infrastructure was initially architected about a decade ago, and the SIF 3.0 infrastructure introduced a wide range of needed new functionality. Of particular note are three ground-breaking design advances which satisfy long standing requests from SIF 2.x developers and implementers. They are summarized below.

Make the SIF Infrastructure independent of the SIF Data Model

All current data model dependencies of the earlier release have been removed. As a result, the SIF 3 infrastructure can carry SIF object data from any locale (US, UK, AU), or for that matter data in conformance with other major data standards, without change.[3] [4]

Offer alternatives and extensions to the required monolithic middleware component (ZIS)

This issue has been addressed on two fronts.

First, the ZIS-provided message broker functionality has been broken up into a set of multiple, separately implementable Infrastructure Services. These were designed to individually map to common industry technologies such as “Enterprise Service Bus (ESB)”, “Queue Manager” and “Service Registry”. The SIF 3 *Brokered Architecture* [5] middleware can still be implemented by a single, monolithic component, but it no longer has to be.

Second, the SIF 3 architecture makes it possible to construct and deploy SIF-compliant solutions in a *Direct Architecture* without utilizing any middleware at all! This new SIF infrastructure alternative offers clients a standardized subset of the functionality available from the *Brokered Architecture*, which means that these clients can be deployed into middleware-centric solutions with no recoding or reintegration efforts required.

Support industry standard transport technologies

The SIF 3 infrastructure documentation describes the service framework and associated core services and utilities in a platform neutral manner. As a result, it can be mapped to any modern transport running over HTTP/S.

The defined platform mapping of the SIF 3 infrastructure is REST. The SIF 3 infrastructure includes paged reads, eTags, synchronous IO, pure data payloads and support for the other primary REST resource design patterns.

---

[3] Yes it’s true. Footnotes in a Read This First! However they will be limited to referencing other parts of the documentation, or to provide information of interest to only a select group of readers.

[4] The SIF Association is currently working with the other major data standards to enable alternative data models to work with SIF 3 infrastructure.

[5] For the definitions of both Brokered and Direct Architectures, please refer to the Glossary provided in the Architecture section of this document.

XPath driven Where Clauses and Named Queries formally define through XQuery are also supported in SIF 3 replacing the earlier SIF-specific *Query* and *Extended Query* functionality.

While this release introduces support for JSON in requests and responses, work continues in order to ensure quality handling of JSON events and add JSON friendly query mechanisms.

These and other changes allow SIF 3 solutions to be deployed in the Data Center of an educational organization using the identical technologies that are already present and known to IT personnel. It also makes it easier for vendors to staff SIF-related projects as both the REST infrastructure technology and to a lesser extent, XPath and XQuery are already familiar to large numbers of today's software developers.

2. Architecture

The most effective path to acquiring a working knowledge of the SIF 3 infrastructure starts with gaining an understanding of the architecture which guided its construction.[6]

This section provides a brief overview of that architecture including:

- Underlying terms and concepts
- Major infrastructure "components"
- Basic functionality
- Supported Message Exchange Patterns (MEPs)
- HTTP Usage

It is the longest section in this RTF.

The reader should have at least a passing understanding of the information presented below before tackling the four volumes of infrastructure documentation.

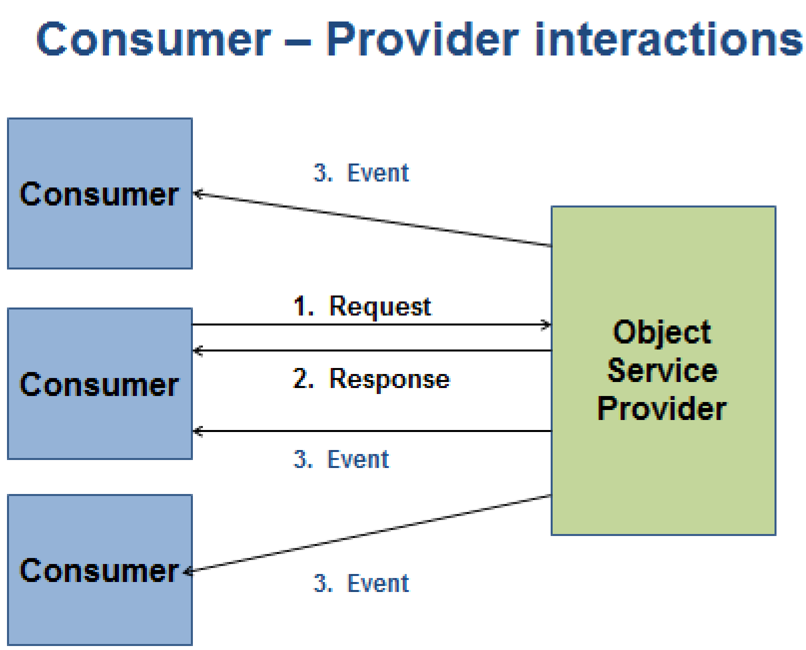
2.1. What are the set of special terms used to describe the infrastructure?

The following entries are abbreviated (rather than rigorous) descriptions selected from the "Glossary of Terms and Concepts" contained in the Base Architecture volume. They comprise the basic terminology – the terms used in the definition of other terms.

Term	Meaning
Object Type	Similar to a "Class" in an object oriented programming language. Every Object Type is defined by its corresponding XML Schema which determines the format of each of its internal elements. Ex: student
Object	An "instantiation" of an Object Type possessing a unique ID, which can be conveyed "over the wire" and validated in accordance with its XML schema. .Ex: student with ID 12345 (John Jones)

[6] *All of the information in this section is provided in greater detail in the SIF 3.2 Infrastructure documentation, and particularly the SIF Base Architecture volume.*

Service Interface	An interface supporting one or more of the standardized CRUD (*Create*, *Query (READ), Update* and *Delete)* data requests on objects of a specific type. Ex: Student Object Service
Event	A message reporting a data change to one or more objects of the same type, often published in response to a successful *Create, Update* or *Delete* data request or action local to the provider. Ex: Student 12345 has updated home phone number: 555-789-4321,
Service Provider	An application providing access to Objects of a selected type in accordance with the Service Interface, and publishing related Events Ex: The Student Service Provider
Service Consumer	Any application which makes requests of, subscribes to, and receives Events from, one or more Service Providers. Ex: A Teaching & Learning cloud application, supporting a course, requesting enrolled students.



2.2. Are Consumers connected directly with Service Providers?

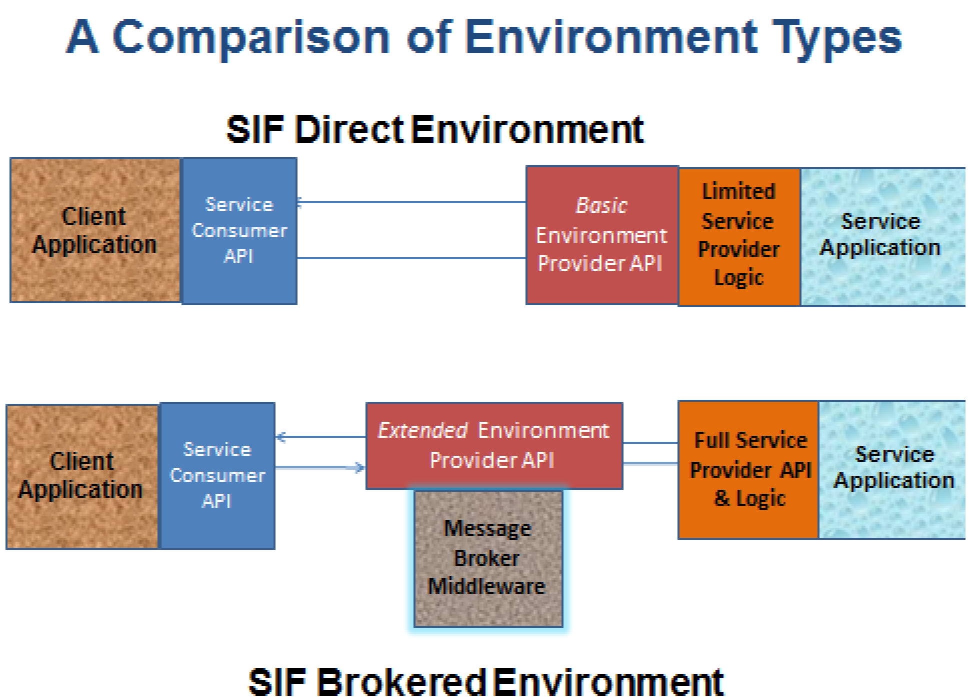
Yes, they may be! There is a third component, called an Environments Provider, which connects the Service Consumers to the Service Providers. However, the Environments Provider **may** be part of the Service Provider and as of SIF 3.1 **may** be ignored by the Service Consumer in many circumstances. Environments are the way "client sessions" are managed, and reflect a considerably more sophisticated approach than traditional REST interfaces. It is the Environment interface that standardizes exactly how SIF 3 data exchanges are guaranteed significant levels of security, enhanced scalability and flexible functionality.

A Consumer's Environment is made available when it initially registers with the Environments Provider Service. It comprises the totality of every possible service the Consumer might ever access. Depending on the authentication provided by the Consumer at registration time, the Environment returned might be testing or staging rather than actual production.

The physical implementation of the Consumer's Environment interface can take one of two forms.

Defined Architectures	Implementation Topologies or Environment Types
Direct	Introduced to the SIF architecture with the 3.0 release, the Direct Architecture standardizes SIF-compliant message exchanges between Consumer and Provider in the absence of a central Message Broker A Direct Architecture *instance* connects a <u>single</u> Service Consumer to a fixed set of one or more directly accessible Service Providers. All Consumer to Provider connections are direct (no middleware components), because the Environments Provider Interface and all Service Provider Interfaces are bundled as a single solution (such as an SIS, LMS, data store, or suite). Such an application generally provides a separate Direct Architecture to each of several Service Consumers, to enable them to directly access and update the object data it provides. In that common case: • Each Service Consumer in a Direct Architecture is operating independently of all other Consumers. • When any Service Consumer request or application action causes a change to the object data maintained in the Service application, every appropriately subscribed Service Consumer receives the identical Event. **Example:** The typical Service Consumer registered in a SIS-provided Direct Architecture could be a simple data entry application running on a mobile device, or a Student Contact system that only needed to access the Student's ID, Name, Addresses and Phone Numbers.
Brokered	The Brokered Architecture securely and reliably connects N Service Consumers to a dynamically changing list of M Service Providers through a Message Broker.

	Unlike the Direct Architecture, any Service Consumer with the proper authorization rights can provision itself as a Service Provider, setting itself up to receive Requests from and publish Events to, other Service Consumers with the appropriate authorization rights. A Brokered Architecture will commonly supply Consumers with access to services providing additional functionality such as the ability to self-provision against a list of provided services. The “Message Broker” functionality requirements of a SIF 3 Brokered Architecture can be implemented (among other alternatives) by SIF “business logic” layered on top of an Enterprise Service Bus (ESB), by internally coupled middleware components or by an upgraded SIF 2.x Zone Integration Server (ZIS). One major feature of this architecture is that the Brokered Architecture offers a superset (rather than replaces) the functionality of the Direct Architecture. As a result, *any Consumer application which interoperates successfully in a Direct Architecture can be redeployed into a Brokered Architecture without reprogramming.* SIF 2.x Brokered Architectures Provider equivalent: ZIS



2.3. How does the Consumer “identify” the Service Provider?

Content-based routing is the ability to route a Request to the appropriate Service instance based upon a specified “service identification”. If the Consumer specifies that service, the

Request will be successfully delivered. In order for this to work, the identification of each Service must be unique.

In the SIF 2.x Zone, every supported object type can have only a single Object Provider. Wherever there are two systems supplying the same type of data (ex: an SIS and a Special Ed application both provide student data within a school), SIF administrators must create two separate Zones. This approach allows the name of the object type provided by a Service to completely identify that Service within the Zone, but often requires a complex N:N mapping of applications to Zones in order to adequately reflect the needs of the educational organization.

With SIF 3, the "Zone" Service Scoping of SIF 2.x is extended both upward (Environment) and downward (Context). Service Identity within an Environment now includes the Zone, the Object Type and the Data Context.

These additional components of this more flexible service identity are individually described in the table below.

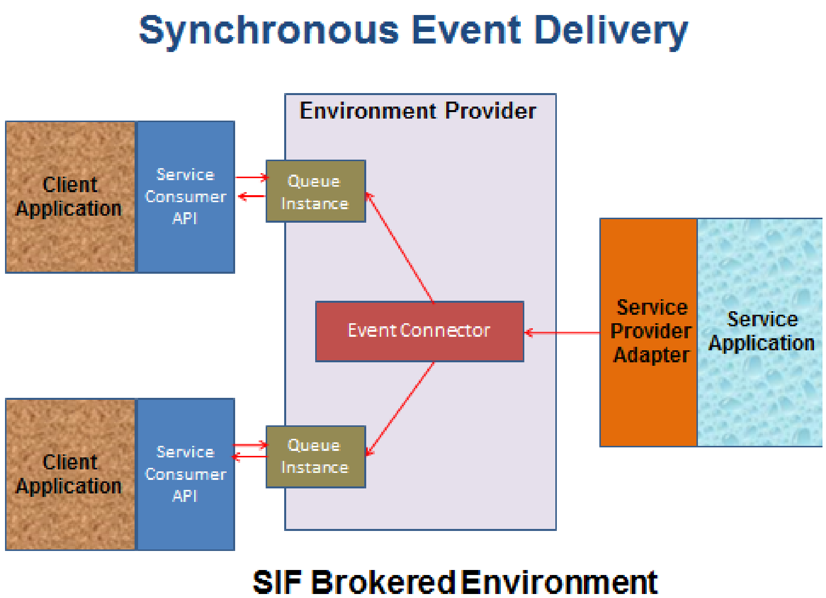
Service Scope Limit	Meaning
Environment	The Environment provided to a Consumer (whether Direct or Brokered) can be divided into multiple Zones, each created by the Environment Administrator to correspond to a physical component within the owning educational organization.
Zone	A Zone scopes a collection of Application Services within the Consumer's Environment. A given Service implementation may support the same Provider interface in several Zones.
Context	A Context is optional Data Model-specific metadata that may accompany a Consumer Request as a way of further scoping and restricting the selected Provider. For example a supplied Context might indicate that the Student Schedule Provider Service being requested in Zone XYZ is the one dealing with next term's data, rather than the current one. A Zone can contain multiple Object Provider Services, each offering its data in a differently named context, or a given Service implementation may support multiple contexts for the same Provider interface. There can be only one Service of a given Object Type with a given Context in any one Zone.

2.4. How does the Provider "alert" the Consumer that an Event occurred?

The publishing of a Service Event (reflecting a change to one or more data objects) is asynchronous from the point of view of a subscribing Consumer since the Service Provider can issue it at any time. The reception of a Service Event is synchronous however. It is deposited into the message queue selected by the Consumer for that purpose when it initially subscribed to change Events for that Object Type, and the Event is later retrieved in response to a Consumer Request to the corresponding Queue Instance Service.[7]

The Service Provider remains unaware of the identity of the particular Consumers who might be subscribed to the Events it issues, and in fact operates independently of the number of subscribers, or even of whether there are any subscribers at all. It publishes all its Events to the Event Connector Service[8], which is responsible for multiplexing the Event message and dispatching a copy of it to the Queue Instance Service of each subscribing Consumer.

The Event delivery process is illustrated in the diagram below.



2.5. How are Provider Responses to Consumer Requests delivered?

In addition to Events, the Queue Instance may also contain delayed Responses to previously issued Consumer Requests. The Consumer specifies in the Request whether the Response is to be immediate or delayed.

---

[7] *For a more complete description of the Queue Service please refer to the appropriate section in the Infrastructure Services document.*

[8] *For a more complete description of the Event Connector Service please refer to the appropriate section in the Infrastructure Services document.*

The details of each mode are described in the table segment below, which is replicated from a larger table in the Base Architecture document.

Response Mode	Details
Immediate	The Response to a Request is provided synchronously in the immediate HTTP response, and the Requester thread for that connection "blocks" until the Response arrives. Immediate Responses are new to SIF 3. They match the standard RESTful Client design pattern and are supported in both Direct and Brokered Architectures.
Delayed	The Consumer issues the Request which is replied to with an "*Accept*" status code in the immediate HTTP response, which indicates "*Request is legal and can and will be delivered to the indicated Service Provider*". This frees a single-threaded Consumer to do other things. The Response issued by the Service Provider arrives asynchronously at a later time, in a manner identical with that of an incoming Event. It contains the Message ID of the original Request it completes.

2.6. How can a Consumer "query" the data maintained by a Provider?[9]

There are several options for the developer of a SIF 3 Service Consumer to choose from. The first is "unqualified" Query, where the Query Responses return all elements from all objects associated with the Query URL.

Query Options	Response
Bulk	All the data for all the objects of the specified type is returned from the Service Provider. As the amount of data may be too large to pack into a single multi-object Response message (causing a Response Error to be generated instead), the Paged alternative shown below is often selected instead.
Paged	A "Paged" Query is typically one in a series of Service Consumer queries for object data, which defines the start and end of the particular Page of results from the Query.

[9] *Please refer to the Query section in chapter 5 of the Base Architecture Guide for a more complete explanation.*

	Example: A teacher holding a tablet device runs a simple Service Consumer REST application that interactively queries the Assessment System for the first 30 Student scores, which are returned immediately. After they are displayed, the teacher may hit "next" whereupon a new interactive Query will be issued, and the next 30 Student scores will be immediately returned and displayed.

There are also three separate and powerful ways to issue Query Requests which impose "constraints" on the object data (Bulk or Paged) returned in the Query Response:

- Use Service Paths to selects the occurrences of one object type in another (ex: *Return every Student in School XYZ).* See section 5.5 of the Base Architecture document
- Use Dynamic Query to specify the "where" criteria which every returned object has to meet (ex: "*Return every Student who is a senior*"). See section 5.7 of the Base Architecture document.
- Use powerful predefined Named XQuerys to select exactly which object elements within each qualifying object of whatever object type (including calculated aggregates) are to be returned. See section 5.56 of the Base Architecture document

2.7. How is HTTP[10] Utilized?

The REST "transport" is for the most part a set of conventions about how to utilize the underlying HTTP line protocol. SIF 3 adopts these conventions and expands them where necessary.

2.7.1. Message Types

The SIF 3 Request / Response types map directly to the common REST usage of the HTTP line protocol:

SIF 3 Request / Response Type	HTTP Message Type
Query	GET
Query Headers	HEAD
Create	POST

[10] *Actually it is more correct to ask how HTTPS is used, as that is the primary line protocol for SIF 3 due to security concerns. However HTTP is optionally supported, and for reasons of clarity will be the protocol referred to.*

Update	PUT
Delete	DELETE

2.7.2. Error Codes

The SIF Error message codes map directly to the HTTP Error codes.[11]

2.7.3. Message Payloads

The payload of every SIF 3 HTTP data message is in XML or JSON format. Specifically:

HTTP Message Type	SIF 3 Message Type	SIF 3 Message Payload
GET	Query Request	Empty.
	Query Response	One or more data objects matching the type of Query Request issued
HEAD	Query Request	Empty.
	Query Response	Empty.
POST	Create Request	One or more data objects, whose elements (possibly excluding the ID, which when missing is assigned by the Service Provider) are provided by the Consumer.
	Create Response	A set of "results" indicating the status of the create for each object
	Create Event	One or more data objects successfully created. The entire object as known to the Provider will be included (i.e. everything returned but non-supported optional elements)
	Update Event	One or more data objects successfully updated. Only those elements actually updated are included, unless the "replacement" flag is set in the HTTP Header, indicating that all the elements are included, whether modified or not.
	Delete Event	The IDs of one or more data objects successfully deleted.
PUT	Update Request	One or more data objects, including the ID, and containing only those elements whose values are to be modified.

[11] *Please refer to Appendix C of the Infrastructure Services document for the complete list of supported HTTP Codes.*

	Update Response	A set of "results" indicating the status of the update for each object
	Multi-object Delete Request	Since REST conventions do not support payloads on HTTP DELETE messages, all multi-object Delete Requests are conveyed via an HTTP PUT message containing an additional HTTP Header Field value of *methodOverride* set to *DELETE.*
	Multi-object Delete Response	A set of "results" indicating the status of the delete for each object
DELETE	Single Object Delete Request	The ID of the single data object to be deleted
	Single Object Delete Response	The "results" indicating the status of the deletion for that object

2.7.4. Non-Payload Information

The "non-object" data associated with these messages (ex: Page Number and Page Size of Query, or Zone and Context of the selected Service Provider) is conveyed in a variety of "non-XML" ways[12].

- Segment in Service URL (Requests only)
- URL Matrix Parameter (Requests only)
- URL Query Parameter (Requests only)
- HTTP Header Field (Requests, Responses and Events)

Every *Resource* (ex: students) may be accessed through one of the following URLs.

- *../resources*: Used for Bulk operations (multi-object change requests)
- *../resources/*12345: Used for single object update or delete of Resource with ID 12345
- *../resources/resource*: Used for single object create of a Resource (at which point the ID is unknown)

The Zone and Context of a Service are indicated in matrix parameters that appear only in the last segment of the Resource URL.

[12] *The definitive sources for the per field mapping of all non-payload information to the URL and HTTP Header fields are Appendix C of the Infrastructure Services document.*

3. SIF 3 Infrastructure Documents and Artifacts

After the basic overview of the SIF 3 infrastructure provided above, it is now possible to present a meaningful list of the various deliverables comprising the SIF 3 Infrastructure release.

3.1. How is the SIF 3 Infrastructure specification documented?

The SIF 3 Infrastructure is described in four "volumes", the first of which (this one) is descriptive, and the last three "normative" in that they define both the mandatory and optional requirements for SIF3-compliant software components

A brief overview of each is volume is contained in the table below.

Infrastructure Volume	Description	Primary Audience
Read This First	The overview, introduction and guide to the other SIF 3 Infrastructure artifacts. It is what you are reading now.	Anyone interested in understanding the functionality of the SIF 3 Infrastructure
Base Architecture	Defines the "core" concepts and detailed service operation framework of the SIF 3 infrastructure, and is the base document on which the other two infrastructure volumes identified below depend.	Those additionally needing to learn about the SIF 3 Infrastructure at a conceptual level.
Infrastructure Services	Defines the complete specification (data structures, operations, and actions) for the set of directly accessible infrastructure services that together comprise the SIF 3 Environments Provider Interface (comparable to the SIF 2.x ZIS interface). This volume details which operations must be supported and which are optional for both the Direct and Brokered Architectures. The most "important Infrastructure Services are: • **Environments:** Authenticates Consumer and defines Consumer Provisioning • **Requests Connector**: Routes all Authorized Consumer Requests to appropriate Service Provider.	Those additionally interested in learning about the core of the SIF 3 Infrastructure at a detailed level, or who need a reference for one or more specific Infrastructure Services.

	• **Queue Instance:** Allows the Consumer to synchronously access asynchronous Responses and Events	
Utilities	Defines the additional set of Services providing secondary infrastructure functionality, which are accessed identically to Object Services. When added to the Infrastructure Services, the set of Service descriptions define the complete SIF 3 infrastructure. The most important Utility Services are: • **Alerts**: Used when creating an exception reporting on an unexpected error condition • **Zones Registry**: Collection of all Zones available to the Consumer application. • **Providers Registry**: Collection of all approved Service Providers available to the Consumer application. Use of this Service is required for all self-provisioning Consumers.	Those interested in learning about the full functionality of the SIF 3 Infrastructure, or who need a reference for one or more specific Utility Services.
Functional Services	Defines the additional infrastructure pieces needed to manage jobs and route messages in-order-to support multiphase services with a beginning, middle, and end. Important infrastructure structures supporting functional services are: • **Job:** Used to track the overall status of a functional service instance. • **State:** Used to set the status of an individual function service phase.	Those interested in defining interaction with more than one step or supporting a predefined functional service.
Data Protection Enforcer Service	Defines optional services for requiring, communicating, and aiding in compliance with Privacy Obligations. Services include: • **Data Privacy Marker:** Used to issue a marker that indicates which Privacy Obligation Document (POD) applies so it can be reused to indicate conformance. • **Filter Request:** Used to aid in conforming to a POD, by reducing the data fields to only those allowed.	Those leveraging interoperability to aid in privacy conformance and enhancement.

	• **Query Validator:** Used to aid in conforming to a POD, by indicating if a given query leverages protected fields or not.	
Version Indication & Negotiation	Adds methodologies to indicate and negotiate not only the version number of the data model being utilized, but also the locale and notation utilized.	Anyone concerned about the format of the data flowing through the infrastructure.

3.2. What purpose do the Infrastructure Object XML Schemas serve?

No surprises here. Like any SIF schema, these explicitly define the format of objects comprising the data payloads of messages exchanged when a Consumer is making requests of a Service. In this case however the Service is either an Infrastructure or Utility Service, and the payload relates to the Infrastructure rather than objects described in the Data Model.

Also as with any XML schema, when provided to an XML programmer toolset they result in the generation of marshaling and unmarshaling stubs that will automatically convert between the XML in a message payload and an "object" as defined in the programming language, and (where validation is enabled) optionally validate that the XML was / is in the correct format.

3.3. What purpose does RIC One Sandbox serve?

Documents and specifications can only go so far. The RIC One Sandbox[13] leverage the Open Framework[14] based upon Java / Swagger technology and was specifically designed to lower the barriers to developing new SIF technology solutions in the following ways:

3.3.1. Learning Tool

RIC One Sandbox provides a comprehensive "hands on" learning tool for SIF xPress developers, featuring a powerful GUI which "*teaches by using*". The GUI functions as a

[13] *Probably the quickest way in North America to come up to speed on the SIF 3 Infrastructure is, after reading this section, go to the RIC One Sandbox URL (*https://sandbox.ricone.org*) and just "explore" with the provided GUI.*

[14] *A great way to build SIF 3 Adaptors Java: https://github.com/Access4Learning/sif3-framework-java .Net: https://github.com/Access4Learning/sif3-framework-dotnet*

Consumer, and communicates with the Direct Architectures Provider component of the Sandbox.

A user can manually generate, view and alter:

- Valid XML instances of key Data Model objects
- The associated URL segments, URL matrix parameters and HTTP header elements used in the construction and deciphering of SIF compliant messages.

3.3.2. REST Platform Mapping Document

A great deal of effort was and will continue to be expended in ensuring that the XML instances and associated URL and HTTP header usage and formats made visible through the Sandbox GUI are valid and in conformance with the Infrastructure documentation.

3.4. What purpose does the SIF 3 Test Harness serve?

The SIF Test Harness provides support for testing: between "*can my software do anything?"* and *"does my software do everything?"* It provides SIF developers with customizable test coverage for the specified range of functionality they believe their application supports, and it forms the basis of the SIF 3 Certification Test Suite.

This allows Developers to focus on the Test Harness when trying to detect problems with their software. When a problem is detected, the Sandbox should be used to isolate the cause and debug the proposed solution.

The SIF 3 Test Harness customizes testing through a document called a Conformance Statement Questionnaire (CSQ). You should consider filling one out before you start your SIF project. This can greatly help with scoping both software or an integration.

3.5. What other resources are available for developers?

There is a growing list of SIF 3 Infrastructure collateral which should be of use to developers.

These currently include:

Resource	Location
A "Member Resources' area on the SIF website has been created specifically to	https://www.A4L.org/page/ResourcesMembers

provide additional resources, documentation and materials. Of specific interest are such topics as: ➢ **SIF 3.0 Support** ○ Common Terms ○ Core Differences between SIF 3.0 and SIF 2.x ○ FAQs ○ 3.0 Resources ○ SIF 3.0 and JSON ○ SIF Environments ○ SIF REST – URL Structure ○ Tools ➢ **xPress Tools and Reference Zone** ○ SIF 3 Concepts ○ SIF 3 & Standards ○ Code Snippets	https://www.a4l.org/page/TechnicalSupport https://xpressapi.org/api-developers/developer-guidance/ https://sandbox.ricone.org/xPress/ (The above URL is for a North American (NA) project)
Official Open Source Resources (developers of SIF 3 solutions will want to consider these)	https://github.com/Access4Learning/sif3-framework-java https://github.com/Access4Learning/sif3-framework-dotnet
Additional SIF 3 infrastructure collateral (presentations, design documentation) available to SIF members only	https://www.a4l.org/page/ResourcesMembers

4. Aspects

This section examines several important aspects of messaging infrastructure in general (modularity, security, reliability and scalability). It highlights the multiple ways in which each aspect is supported by the SIF 3.0 Infrastructure, and provides references to the release documentation that describes the aspect in further detail.

Those readers desiring an understanding of one (or more) of these aspects as it relates to the SIF 3.0 Infrastructure should read the relevant subsection(s) below.

4.1. How modular are SIF 3 Environments?

A Consumer of a Service can be written to support only the functionality it needs. The situation is more complex for Environments Service Providers because the developers must correctly anticipate the level of functionality deemed critical to the set of applications registering themselves in the Environments Service Provider. If no guidelines were specified, each Environments Service implementation could support a different combination of features, significantly reducing “out of the box” interoperability between SIF-conformant applications.

The SIF 3 Infrastructure standard addresses this problem by defining a small set of “Environment Service functionality packages” of increasing breadth and power. Each package consists of a set of required functionality that an Environments Provider could certify to, although a given product could optionally provide additional functionality as well.

For example, the requirements placed on an Environments Provider at a given SIF 3 site could range between supporting:

- A Direct Architecture implemented by an SIS which services multiple instances of the same REST-based Dashboard Consumer application, where each instance runs on a different remote tablet device and each issues only a series of interactive Page *query* requests for Student data.

- A state-wide Brokered Architecture implemented by extensions to an ESB middleware, product, which connects multiple regional and district SIS application Consumers dynamically issuing CEDS-compliant *create* and *update* Student data requests to the organization’s SLDS, and optionally receiving published Events describing all data changes.

In the SIF 3 architecture, the service functionality required of the first Environment is a completely contained subset of the service functionality required of the second. As a result, developers of the dashboard application can focus only on the manageable subset of the infrastructure documentation, which relates to their specific needs, and yet still have their application work identically in both types of Environments.

4.2. How Secure are SIF 3 solutions?

As in previous releases, SIF 3 was designed with Data Security baked in rather than being added on at the end. The Brokered Architecture is essentially a "Data Confederacy" where data owners maintain control of their data and must "opt in" to share their data with other entities (application or organizational) *rather than* a "Data Union" with a single component that every application **must** share its data with.

At any site where SIF 3 is deployed, it is therefore the local data administrators who determine which applications (local and remote) can access or update sensitive data such as student discipline or health information, and exactly which parts of that data will be made visible.

The following table highlights some of the major security functionality provided / supported by the SIF 3 Infrastructure, and indicates where it is described in the specification.

Security Functionality	Description	Reference
Authentication	The way in which the recipient of a message confirms the identity of the sender. In SIF 3, this occurs when an application initially registers as a Consumer with the Environments Provider, or when a Consumer issues a Request or a Service publishes an Event to the appropriate Connector. A Consumer's identity is a combination of Application, Application Instance and/or User information. Multiple authorization methods involving a "shared secret" are defined. Also considerations for SSO technologies have been made.	The *Environments Service* section in Infrastructure Services.

Authorization	The way in which a Service Provider, having authenticated the identity of the Consumer issuing a Request, determines what access rights that Consumer has been given, and the ways in which limitations on those rights can be enforced. Site administrators can fine tune which Object Services a specific Consumer can access, and what types of access will be allowed	The *Environments Service* and *Provision Request Service* sections of Infrastructure Services.
Encryption	The way in which the contents of a message being routed over an insecure network is protected against being intercepted and understood by a 3rd party.	Enabled but not standardized. As stated: *The Provider should use modern libraries and frameworks in order to support the likely high levels of encryption keys and certificate trust scrutiny; however the details of such support are no longer conveyed within the SIF standard* See the SIF 3 Product Standard[15] for additional certification requirements.
Content Accountability	Determining responsibility for data changes. In SIF 3 every element in a message payload can be traced back to the issuing application, and optionally, to the actual person responsible for the current value.	The *generatorId* element as described in the Base Architecture.
Privacy	This infrastructure contains services to: disclose privacy obligations, issue markers indicating the set of privacy obligations, enforce the use of privacy markets, and aid in privacy enforcement.	The Privacy Obligations service can be found with the Utility Services and its payload is define in the Specification & Schemas. All other privacy services can be found in the Data Protection Enforcer Service volume.

[15] *http://cert.sifassociation.org/Shared%20Documents/SIF%203%20Product%20Standard.pdf*

4.3. How reliable are SIF 3 Solutions?

Overall system reliability is of course dependent on the quality of the set of applications deployed in a given solution, and the extent to which these applications successfully interact.

The SIF Certification program provides the primary way to ensure seamless interoperability between SIF-certified components.

In addition, the SIF 3 architecture defines and standardizes a framework for constructing reliable multi-application solutions. The table below highlights the main components of this framework.

Reliability Framework Component	Description	Reference
Guaranteed Message Delivery	The SIF standard requires that any issued message (Request, Response or Event) arrive at its destination, or (in unrecoverable error situations) that appropriate notification of the problem be given. In a Brokered Architecture, this means a Service Provider can publish an Event, and it will eventually be delivered to every authorized subscriber, even if the recipient was off line or unavailable when the Event was posted.	The *Request / Response / Event Message Exchange Choreography* section in the Base Architecture.
Error Handling Analysis	This analysis standardizes the process description of exactly how to handle errors detected in any part of the message exchange process, for Requests, Responses and Events.	The *Request / Response / Event Message Exchange Choreography* and the *Error Handling* sections in *the Base Architecture.*
Alerts Service	The *Alerts* Utility Service provides a way for Consumers and Service providers to create an Alert object describing an exception condition that has just occurred. This could range from receiving an Event payload that failed to validate, to receiving an *update* request for an object that does not exist. The implementation of *Alerts* may range from appending each created Alert to an Error Log file for later examination by site management, to converting	The *Alerts Service* section in the Utility Services document.

	and forwarding the Alert to a Preventative Maintenance and / or Service Management system.	
Error Object	An Error object may be contained in place of one or more object results in the Response to an erroneous Consumer Request. The value of the internal Error code element matches one of the set HTTP error codes utilized by REST programs, and the remainder of the Error object expands on the reason.	The *Error Handling* section in *the Base Architecture.*

4.4. How Scalable are SIF 3 deployments?

SIF 2.x Zone-based solutions have been deployed in a wide variety of educational organizations ranging from a single school to an entire State, for a wide variety of purposes. Performance limits were encountered at several particularly large deployments, especially at the end of a reporting period when message traffic volumes soared.

A careful analysis was made of the architectural causes of these limits, and all of the identified factors were addressed. There is excellent reason to believe that performance will be substantially improved for the entire SIF 3 lifecycle.

The following table highlights some of the major performance enhancements contained in the SIF 3 Infrastructure, and indicates where they are described in the specification.

Performance Enhancement	Description	Reference
Multiple Consumer Sessions for same application	Site Administrators can enable the same application instance, using multiple *identifier* values (ex: *Johnson JHS* and *Miller Elementary*) to register itself as multiple completely independent and parallel Consumers.	The *Environments Service* "create Environment" section in Infrastructure Services.
Multiple FIFO Message Queues for same Consumer	Any Consumer can set up separate Message Queues and assign different Event types to them. This allows the less important Events (ex: Attendance) to sit in Queue B until the messages in Queue A have all been retrieved and processed.	The *Environment Scalability* section in Infrastructure Services.

Multiple concurrent Consumer connections option to same Queue	A Consumer may use (where the Queue Service supports it) more than one multiple simultaneous concurrent connections, having multiple HTTP Requests open with a single Queue.	The *Queue Service* section in Infrastructure Services.
"*Get Next and Pop*" Queue Service Query Request	This replaces the SIF 2.x "double handshake" between Consumer and Polling Queue with a single handshake, reducing the number of message exchanges necessary to retrieve an incoming Event during periods of high message traffic by a factor of two.	The *Queue Service* section in Infrastructure Services.
"Long Polling"	This option eliminates the message retrieval latency time previously associated with repeated polling for an arriving message, in periods of light traffic.	The *Queue Service* section in Infrastructure Services.
Multiple Objects conveyed in single Request and single Event	The ability to "pack" a single Request to or Event with multiple affected objects from a specific Service Provider offers huge performance benefits for several important use cases (ex: a *thousand small Attendance Objects created at the end of a Reporting Period result in the generation of 10 multi-object Events rather than 1000 single object Events)*.	The *Service Operation* section in the Base Architecture.
Immediate Service Responses	In another major change from SIF 2.x, a Consumer may now request that the Service Response to any given Request be returned synchronously (i.e. immediately) on the HTTP Response to its issued HTTP Request, rather than appear asynchronously at some later time, intermixed with arriving Events. This will likely be the common case for most Requests, it conforms to the standard REST design pattern, it maximizes the effectiveness of Direct Architectures and it is expected to improve overall Request / Response processing throughput significantly.	The *Basic Infrastructure Framework* section in the Base Architecture.
eTag	An eTag is equivalent to a "checksum" on all the objects of the type being queried which are maintained by the Service Provider, and it is optionally returned in all Query Responses.	The *Service Operation (Query)* section in the Base Architecture.

	If a valid eTag is included in a Query Request, and if there were no object data changes since the eTag was created, no objects are returned. This offers an efficient way for Consumer applications executing on mobile devices (which are unlikely to subscribe to Events) to determine whether they already have the latest data on one or more objects without comparing all the elements.	

Appendix A: Miscellaneous Questions

The following questions did not fit into any specific technical section, but the answers were felt to be of general interest.

A.1 Isn't SIF an overly complex infrastructure specification?

The documentation for all 6 volumes of the SIF 3 Infrastructure specification totals around 250 pages. The primary reason for this was the need to standardize how the extensive level of functionality required to support large State Level educational solutions would be provided.

Packages within Packages

Sometimes it takes a lot of careful design to make something simple.

SIF 3 functionality has been specified as a modular collection of RESTful service interfaces, packaged together in one of three layers of increasing power, each building on the last. Since each functionality layer is a completely contained subset of the one "higher up" in the functionality strata, application developers need to understand and utilize only that subset of the infrastructure documentation that relates directly to their needs. Their resulting applications will work identically in any deployment supporting at least the set of functionality in the package they are designed to, or any level above that.

Leveraging Infrastructure Vendor Offerings

It is also anticipated that in a manner similar to how earlier releases evolved, much of the SIF 3-specific infrastructure support will likely be provided by dedicated SIF infrastructure product vendors or open source SIF-adapter projects. This will leave many educational application developers free to focus primarily on the functionality of their product and how it bridges to the relevant elements of the SIF Data Model version which they support, rather than on the details of the infrastructure over which their payloads travel.