HL7 Version 3 Standard: 
Retrieve, Locate and 
Update Service (RLUS), 
Release 1 

July 2019

Service Functional Model Specification

Reaffirmation of ANSI/HL7 RLUS, R1-2013

Sponsored by:
Services Oriented Architecture Work Group

Services Oriented Architecture Work Group Co-Chairs:

Don Jorgenson

Vincent McCauley, MB BS, PhD

Medical Software Industry Association (Australia)

Ken Rubin

Hewlett-Packard Enterprises Services
IMPORTANT NOTES:

HL7 licenses its standards and select IP free of charge. **If you did not acquire a free license from HL7 for this document**, you are not authorized to access or make any use of it. To obtain a free license, please visit http://www.HL7.org/implement/standards/index.cfm.

If you are the individual that obtained the license for this HL7 Standard, specification or other freely licensed work (in each and every instance "Specified Material"), the following describes the permitted uses of the Material.

A. HL7 INDIVIDUAL, STUDENT AND HEALTH PROFESSIONAL MEMBERS, who register and agree to the terms of HL7’s license, are authorized, without additional charge, to read, and to use Specified Material to develop and sell products and services that implement, but do not directly incorporate, the Specified Material in whole or in part without paying license fees to HL7.

B. HL7 ORGANIZATION MEMBERS, who register and agree to the terms of HL7’s License, are authorized, without additional charge, on a perpetual (except as provided for in the full license terms governing the Material), non-exclusive and worldwide basis, the right to (a) download, copy (for internal purposes only) and share this Material with your employees and consultants for study purposes, and (b) utilize the Material for the purpose of developing, making, having made, using, marketing, importing, offering to sell or license, and selling or licensing, and to otherwise distribute, Compliant Products, in all cases subject to the conditions set forth in this Agreement and any relevant patent and other intellectual property rights of third parties (which may include members of HL7). No other license, sublicense, or other rights of any kind are granted under this Agreement.

C. NON-MEMBERS, who register and agree to the terms of HL7’s IP policy for Specified Material, are authorized, without additional charge, to read and use the Specified Material for evaluating whether to implement, or in implementing, the Specified Material, and to use Specified Material to develop and sell products and services that implement, but do not directly incorporate, the Specified Material in whole or in part.

Legimate use of Specified Material is subject to the full license terms governing the Material. Please see http://www.HL7.org/legal/ippolicy.cfm for the full license terms governing the Material.

Ownership. Licensee agrees and acknowledges that HL7 owns all right, title, and interest, in and to the Trademark. Licensee shall take no action contrary to, or inconsistent with, the foregoing.

Licensee agrees and acknowledges that HL7 may not own all right, title, and interest, in and to the Materials and that the Materials may contain and/or reference intellectual property owned by third parties (“Third Party IP”). Acceptance of these License Terms does not grant Licensee any rights with respect to Third Party IP. Licensee alone is responsible for identifying and obtaining any necessary licenses or authorizations to utilize Third Party IP in connection with the Materials or otherwise. Any actions, claims or suits brought by a third party resulting from a breach of any Third Party IP right by the Licensee remains the Licensee’s liability.

Following is a non-exhaustive list of third-party terminologies that may require a separate license:

<table>
<thead>
<tr>
<th>Terminology</th>
<th>Owner/Contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNOMED CT</td>
<td>SNOMED International; <a href="http://www.snomed.org/snomed-ct/get-snomed-ct">http://www.snomed.org/snomed-ct/get-snomed-ct</a> or <a href="mailto:info@ihtsdo.org">info@ihtsdo.org</a></td>
</tr>
<tr>
<td>Logical Observation Identifiers Names &amp; Codes (LOINC)</td>
<td>Regenstrief Institute</td>
</tr>
<tr>
<td>International Classification of Diseases (ICD) codes</td>
<td>World Health Organization (WHO)</td>
</tr>
<tr>
<td>NUCC Health Care Provider Taxonomy code set</td>
<td>American Medical Association; Please see <a href="http://www.nucc.org">www.nucc.org</a>. AMA licensing contact: 312-464-5022 (AMA IP services)</td>
</tr>
</tbody>
</table>
Preface

Acknowledgements

Project Lead/Editor  
Stefano Lotti, HL7 Italy - Invitalia

Contributors  
John Koisch, Guidewire Consulting  
Ken Rubin, HP Enterprise Services  
Alan P. Honey, Kaiser Permanente  
Scott Robinson, Kaiser Permanente  
Kensaku Kawamoto, University of Utah

Notes to Readers

This document is the Service Functional Model (SFM) for the Retrieve, Locate, and Update Service, which is specified under the Service Development Framework process under the auspices of the Healthcare Services Specification Project (HSSP)

Further context is given in the overview section below, but one key point to note is that the SFM provides a Service Interface specification\(^2\) at a functional level, NOT the specification of a Service implementation. The SFM described here is complemented by a Service Technical Model (STM) that specifies the technical requirements of the service\(^3\).

Some complementary material (not normative) to this specification, as UML models source files, can be found at the RLUS wiki site\(^4\).

Changes from Previous Release

This document is an evolution of the previous DSTU document and it’s based on the lesson learned from the Service Technical Model (STM) specification from OMG and other experiences of SFM use in real scenarios. The main change is a more clear independence of RLUS from an implementation architecture. Now a registry and repository architecture deployment is not required and is only an option. The new version, in alignment with SAIF, makes extensive use of modeling languages as UML 2.x\(^5\), BPMN 2.0\(^6\) and SoaML 1.0\(^7\).

---

1. See: http://hssp.healthinterop.org
2. The term “Interface specification” is intended within the service orientation context, so it’s not restricted to user interface. See Glossary (Appendix A) for the definition of the “Interface Specification” term.
5. See: http://www.uml.org/
6. See: http://www.omg.org/spec/BPMN/2.0/
7. See: http://www.omg.org/spec/SoaML/
Acknowledgements

RLUS represents a considerable undertaking by members of the Healthcare Service Specification Project. Many thanks to the members of the HSSP team who have contributed time and effort to this specification. In particular, Andy Bond and Ken Rubin have greatly supported the final revision of the document.

As usual, the HSSP team has proven to be a valuable, effective community to find solutions in this space. Thanks to everyone for another successful effort.

Table of Contents

TABLE OF CONTENTS .................................................................................................................................4

LIST OF DIAGRAMS .......................................................................................................................................8

1. EXECUTIVE SUMMARY ..............................................................................................................................10
   1.1 Service Overview ......................................................................................................................................10
   1.2 Scope .....................................................................................................................................................11

1.3 Assumptions and Dependencies ..................................................................................................................11
   1.3.1 Service Orientation ............................................................................................................................12
   1.3.2 Semantic Signifier ..............................................................................................................................12

2. IMPLEMENTATION CONSIDERATIONS ....................................................................................................14
   2.1 Deployment Scenarios ............................................................................................................................15
      2.1.1 Intra-organizational deployment .......................................................................................................16
      2.1.2 Cross-organizational deployment: centralized ................................................................................16
      2.1.3 Cross-organizational deployment: registry and repository architecture ...........................................17
      2.1.4 Cross-organizational deployment: peer to peer ..............................................................................17

3. BUSINESS CONTEXT ..................................................................................................................................19
   3.1 The reason why the service is necessary .................................................................................................19
   3.2 Business Scenarios ..................................................................................................................................19
      3.2.1 Scenarios overview ...........................................................................................................................19
      3.2.2 Scenario Actors ..................................................................................................................................20

3.3 Core Runtime Scenarios ............................................................................................................................21
   3.3.1 Locate Resources ..................................................................................................................................21
      3.3.1.1 Scenario .........................................................................................................................................21
      3.3.1.2 Interfaces Used .............................................................................................................................21
      3.3.1.3 Scenario Details .............................................................................................................................21
   3.3.2 Get Resource ........................................................................................................................................22
      3.3.2.1 Scenario .........................................................................................................................................22
      3.3.2.2 Interfaces Used .............................................................................................................................22
3.3.2.3 Scenario Details .................................................................................................................................. 22
3.3.3 List and Get Resources ............................................................................................................................ 23
  3.3.3.1 Scenario ................................................................................................................................................ 23
  3.3.3.2 Interfaces Used ..................................................................................................................................... 23
  3.3.3.3 Scenario Details .................................................................................................................................. 23
3.3.4 Put Resource ............................................................................................................................................ 24
  3.3.4.1 Scenario ................................................................................................................................................ 24
  3.3.4.2 Interfaces Used ..................................................................................................................................... 24
  3.3.4.3 Scenario Details .................................................................................................................................. 24
3.3.5 Initialize Resource .................................................................................................................................... 25
  3.3.5.1 Scenario ................................................................................................................................................ 25
  3.3.5.2 Interfaces Used ..................................................................................................................................... 25
  3.3.5.3 Interaction Details ................................................................................................................................. 25
3.3.6 Discard Resource ...................................................................................................................................... 26
  3.3.6.1 Scenario ................................................................................................................................................ 26
  3.3.6.2 Interfaces Used ..................................................................................................................................... 26
  3.3.6.3 Scenario Details .................................................................................................................................. 26

3.4 Administrative Scenarios ............................................................................................................................. 27
  3.4.1 Link Resources .......................................................................................................................................... 27
    3.4.1.1 Scenario ................................................................................................................................................ 27
    3.4.1.2 Interfaces Used ..................................................................................................................................... 27
    3.4.1.3 Scenario Details .................................................................................................................................. 27
  3.4.2 Unlink Resources ....................................................................................................................................... 28
    3.4.2.1 Scenario ................................................................................................................................................ 28
    3.4.2.2 Interfaces Used ..................................................................................................................................... 28
    3.4.2.3 Scenario Details .................................................................................................................................. 28
  3.4.3 Merge Resources ....................................................................................................................................... 28
    3.4.3.1 Scenario ................................................................................................................................................ 28
    3.4.3.2 Interfaces Used ..................................................................................................................................... 29
    3.4.3.3 Scenario Details .................................................................................................................................. 29
  3.4.4 Unmerge Resources ................................................................................................................................... 29
    3.4.4.1 Scenario ................................................................................................................................................ 29
    3.4.4.2 Interfaces Used ..................................................................................................................................... 30
    3.4.4.3 Scenario Details .................................................................................................................................. 30

3.5 Introspective Scenarios ................................................................................................................................. 30
  3.5.1 List and Describe Semantic Signifiers and Conformance Profiles ............................................................... 30
    3.5.1.1 Scenario ................................................................................................................................................ 30
    3.5.1.2 Interfaces Used ..................................................................................................................................... 31
    3.5.1.3 Scenario Details .................................................................................................................................. 31
  3.5.2 Put Semantic Signifier ............................................................................................................................... 31
    3.5.2.1 Scenario ................................................................................................................................................ 31
    3.5.2.2 Interfaces Used ..................................................................................................................................... 31
    3.5.2.3 Scenario Details .................................................................................................................................. 32

4. DETAILED FUNCTIONAL MODEL .................................................................................................................... 33

4.1 Structure of the Service .................................................................................................................................. 33
4.2 RLUS Conformance........................................................................................................................................ 36

4.3 Assumptions and Dependencies.................................................................................................................. 37
  4.3.1 Assumptions................................................................................................................................................ 37
    4.3.1.1 Resource Identification and Location .................................................................................................. 37
    4.3.1.2 Entity Identification ............................................................................................................................ 37
    4.3.1.3 Information Quality ............................................................................................................................ 38
  4.3.2 Dependencies........................................................................................................................................... 38
    4.3.2.1 Dependencies on other HSSP Infrastructure Services ......................................................................... 38
    4.3.2.2 Cross-organizational RLUS and Discovery and Description Services .............................................. 38
    4.3.2.3 Support for Semantic Signifiers ........................................................................................................... 38

4.4 Basic Run-Time Capabilities.......................................................................................................................... 39
  4.4.1 Locate Resources ....................................................................................................................................... 39
  4.4.2 Get Resource ............................................................................................................................................. 40
  4.4.3 List and Get Resources .............................................................................................................................. 41
  4.4.4 Put Resource ............................................................................................................................................ 41
  4.4.5 Initialize Resource ..................................................................................................................................... 42
  4.4.6 Discard Resource ....................................................................................................................................... 43

4.5 Administrative Capabilities............................................................................................................................ 43
  4.5.1 Link Resources .......................................................................................................................................... 43
  4.5.2 Unlink Resources ....................................................................................................................................... 44
  4.5.3 Merge Resources ....................................................................................................................................... 45
  4.5.4 Unmerge Resources .................................................................................................................................. 45

4.6 Introspective Capabilities.................................................................................................................................. 46
  4.6.1 List Conformance Profiles ......................................................................................................................... 46
  4.6.2 List Semantic Signifiers ............................................................................................................................. 47
  4.6.3 Describe Semantic Signifier ..................................................................................................................... 48
  4.6.4 Put Semantic Signifier .............................................................................................................................. 48

5. PROFILES...................................................................................................................................................... 49
  5.1 Introduction .................................................................................................................................................. 49
  5.2 Functional Profiles ..................................................................................................................................... 50
  5.3 Semantic Profiles and Semantic Signifier .................................................................................................... 51
  5.4 Conformance Profiles .................................................................................................................................. 52

APPENDIX A: GLOSSARY ........................................................................................................................................ 54

APPENDIX B: PROFILES EXAMPLES ................................................................................................................. 57

APPENDIX B.1: CDAR2 PROFILES ...................................................................................................................... 57

APPENDIX B.2: RLUS REGISTRY REPOSITORY PROFILE WITH RCMR AND CDAR2.............. 59
APPENDIX C: HL7 EHR FUNCTIONAL MODEL TRACEABILITY .................................................................61

APPENDIX D: MAPPING WITH SERVICE TECHNICAL MODEL (STM) ..............................................68

APPENDIX E: RELEVANT STANDARDS AND PROFILES ...................................................................71
List of Diagrams

Figure 1: Representative deployment of RLUS within an organization (intra-organizational) ........................................ 15
Figure 2: Representative RLUS deployment in a cross-organizational setting (centralized) ........................................ 16
Figure 3: Representative RLUS deployment in a registry and repository scenario ................................................... 17
Figure 4: Representative RLUS deployment in a cross-organizational p2p setting (inter-organizational) .................. 18
Figure 5: Locate Resource scenario .......................................................................................................................... 22
Figure 6: Get Resource Scenario .............................................................................................................................. 22
Figure 7: List and Get Resource Scenario .................................................................................................................. 23
Figure 8: Put Resource Scenario ............................................................................................................................... 24
Figure 9: Initialize Resource Scenario ....................................................................................................................... 25
Figure 10: Discard Resource Scenario ..................................................................................................................... 26
Figure 11: Link Resource Scenario ............................................................................................................................ 27
Figure 12: Unlink Resource Scenario ......................................................................................................................... 28
Figure 13: Merge Resource Scenario ........................................................................................................................ 29
Figure 14: Unmerge Resource Scenario .................................................................................................................... 30
Figure 15: List and Describe Semantic Signifiers Scenario ....................................................................................... 31
Figure 16: Put Semantic Signifier Scenario ............................................................................................................ 32
Figure 17: RLUS Runtime Capabilities .................................................................................................................... 34
Figure 18: RLUS Administrative Capabilities ........................................................................................................... 35
Figure 19: RLUS Introspective Capabilities ................................................................................................................ 36
Figure 20: Abstract Structure of a Semantic Signifier .............................................................................................. 52
Figure 21: Abstract Structure of a Conformance Profile .......................................................................................... 52
Figure 22: CDA 2 R-MIM ........................................................................................................................................... 57
Figure 23: RLUS Gateway Services Architecture with a CDAr2 profile ................................................................. 58
Figure 24: RCMR D-MIM ........................................................................................................................................ 59
Figure 25: RLUS Registry and Repository Services Architecture .............................................................................. 60
Figure 26: RLUS STM PIM Service Architecture ................................................................. 68
Figure 27: RLUS STM PIM Services Contracts ................................................................. 69
Figure 28: RLUS SFM (CIM)-STM (PIM) Mapping ............................................................ 70
21. Executive Summary

31.1 Service Overview

The Retrieve, Locate, and Update Service (RLUS) Service Functional Model specification provides a set of capabilities through which information systems can access and manage information resources.

RLUS realizes, at its core, a basic set of CRUD\(^8\) capabilities plus location for health information resources management and, simply, standardizes the way in which the resources are exposed and consumed independently from the nature of the resources.

RLUS explicitly occupies the service space within an information processing environment. It is independent but compatible with underlying structures, including local security implementations, data models, or delivery mechanisms. By separating and exposing those aspects of resources that facilitate inter-organization workflows in a service layer, this specification abstracts the problem of interoperability away from underlying systems. This level of abstraction and reconfiguration allows interoperability and system durability and reduces burdensome technology integration.

HL7 International is participating in a multi-organizational collaboration called the Healthcare Services Specification Project (HSSP)\(^9\) in which these activities are occurring.

The business cases supported by RLUS are common to healthcare, but frequently are implemented in ad hoc ways and with platform dependent issues. These business cases generally involve multiple partners in the health team sharing data over long periods of time. RLUS provides an appropriate space to define and implement these business cases with the overall effects on internal systems minimized.

In RLUS the nature of the resources is managed by means of the implementation of the Semantic Signifier concept\(^10\). The Semantic Signifier defines a mechanism to abstract the RLUS capabilities from different kinds of resources and data structure that can be managed in a flexible and standardized way.

From the business viewpoint this is relevant because an RLUS interface can be reused in several contexts and with different data formats and can evolve easily in response of specific business change within a context.

As a consequence of its service oriented approach we can understand the relevance of the RLUS standard if we consider not only the specification in itself as a means to define the responsibility of a RLUS compliant system. We should consider the RLUS specification in a real enterprise service architecture where flexibility, adaptability and governance are critical requirements.

---

\(^{8}\) Create, Read, Update and Delete.

\(^{9}\) See: http://hssp.healthinterop.org

\(^{10}\) See § 1.3.2 and § 5.3 and Glossary (Appendix A) for definition and characteristic of Semantic Signifier.
311.2 Scope

The Retrieve, Locate, and Update Service (RLUS) Service Functional Model specification seeks to define, at a service level, the appropriate capabilities that must be realized by an interface to locate, retrieve, and update resources among healthcare organizations.

RLUS provides architects and designers with an effective mechanism to specify, scope, and implement the portions of a healthcare enterprise architecture committed to information object interoperability. Whether this interoperability is among departments, systems, or organizations, RLUS embodies a flexible, responsive interface standard that shortens design time and reduces product iterations.

An RLUS occupies the portion of the technology stack that is dedicated to the service space. As a service or a set of services, RLUS provides for low-cost, easily maintainable components that can be modelled to fit most deployment scenarios. Without specifying particular delivery mechanisms, payloads, or technologies, a clear strategy that lays the groundwork for bridging the gap between information systems is nevertheless outlined.

RLUS, it’s only a service specification so it’s not intended to replace existing systems or implementations, but to create an interface standard for a service-oriented layer to expose those healthcare assets and resources within an organization, or among organizations, that need to meet business and/or clinical needs.

One of the chief methods to this end is the specification of RLUS Conformance Profiles. Conformance Profiles allow an organization to align with different trading partners according to different trading rules, as realized by functional and semantic profiles. Simple RLUS interfaces may conform to different deployment scenarios and different conformance profiles while still using the same structures and interfaces.

Compliance to the RLUS Functional Model provides the fundamentals for implementing an organization’s business strategies that extend beyond core business components. Interoperability services become an effective means by an organization to meet the shifting industry or trading requirements that characterize the healthcare industry. As such, RLUS compliant services reduce implementation costs for an organization while serving to meet the current and evolving interoperability strategies imposed both from above and below.

The manner in which services and interfaces are deployed, discovered, etc. is outside the scope of this functional model. These aspects are left to a conformant Service Technical Model and at a number of implementation guides that will be made available over time. All other interactions within the scenarios identified in Section § 2 are considered in scope.

As that specified in OMG Retrieve, Locate, and Update (RLUS) Service (see: http://www.omg.org/spec/RLUS/) based on WS* stack or the emergent hData RESTful implementation.
1.3.1 Service Orientation

RLUS supports in its design the classical principles of service orientation as standardization of service contracts, loose coupling, service abstraction, reusability, autonomy, service stateless, discoverability, and composability.

RLUS is autonomous and decoupled from implementation details. Programmatically RLUS services can be used in a pre-existing implementation or can serve as an interface specification for a new RLUS implementation.

No reference to internal technical characteristics exist in a conformant RLUS specification.

RLUS excludes from its scope some interoperability components whose functions are assumed to either exist or be provided elsewhere, including security, terminology resolution, identity management, underlying data structures, or specific implementations of record storage, registry, or creation.

While RLUS maintains no formal dependencies on these areas, RLUS implementations may elect to take advantage of their existence when those functions are available in a Service Oriented Architecture. It is important to understand that these kinds of dependencies are exclusively business lead and don't break RLUS autonomy.

HSSP produces a growing Service Inventory of useful healthcare standard services that can be used in conjunction with RLUS. For example, a cross-organizational RLUS takes advantage on a cross domain entity identification service, which might conform to the HSSP's IXS specification or, in a same way, a specific RLUS implementation scenario can have a dependence on a Common Terminology Service Release 2 (CTS2).

Without doubt the existence of a set of consistent services can be useful in a real world implementation and, in this sense, the full potential of a RLUS implementation is realized when deployed as a member of a standardized service architecture but is not tied to RLUS in itself.

1.3.2 Semantic Signifier

A design goal of RLUS is its flexibility and adaptability to different semantic content. The way to realize it is the separation of concern between functionality and semantic content in the RLUS interface. So a relevant concept assumed in RLUS is the Semantic Signifier.

The purpose of the Semantic Signifier is to separate the generic functional capabilities of RLUS from the semantic content. In this way RLUS can support different information objects and different metadata sets in a flexible but standardized and consistent interface.

Semantic Signifiers facilitate a meaningful interchange of information between transactions involving RLUS. RLUS stops short, however, of mandating the specific information content to be carried by the service for several reasons. First, this provides the implementer the ability to use the information semantics that are most appropriate for their needs. While infrastructure is being put into place for industry consensus-building around standard Semantic Signifiers (and implementers are encouraged to use them), this is not mandated.

Semantic Signifiers within the base specification are expressed as a datatype allowing a platform-level binding of RLUS, while keeping open a construct allowing for scalability, extensibility, and diversity of semantic information. Additionally, RLUS subscribes to the HSSP’s profiling mechanism to allow for strong conformance assertions to be made, inclusive of informational semantics and designated semantic signifiers. This approach allows RLUS to carry payloads that have been standardized by other specifications or groups.

The structure defined by Semantic Signifier is used to specify the different types of content that can be conveyed by an RLUS interface and at the same time it’s the reference for the RLUS query mechanisms (by parameter or by example).

Concretely an HL7 Semantic Signifier can be generic CDA R2 or can be derived from a CDAr2 Implementation Guide (as a Continuity of Care Document or IHE XD-LAB Laboratory Report) as well as from HL7-v3 Domain Message Models (D-MIM) or directly from Refined Message Information Models (R-MIM). In RLUS the query mechanisms are on “the skin” of the interface without any reference to an underlying implementation. Any detail about the internal implementation is excluded (e.g. no stored query reference, no underlying database structure, programming language, platform, etc.).

At run-time, a semantic signifier is the mechanism for realizing semantic profiles. This element may be for example an HL7 artifact, a locally used template, a nationally published template, a local XML schema, or any other agreed upon set of values. It may be passed by reference or by value, as either may satisfy the functional requirements and meet the business need. However, it is vital that a deployed RLUS can describe information about the semantic profile or profiles that it supports. Depending on the functional profile supported in a deployment, this could include:

- that it specifically delineates the semantic signifiers by which queries may be made
- that it specifically delineates the semantic signifiers by which responses will be delivered
- that it supports queries of one resource from another resource where both are described by semantic signifiers.

Semantic transformation or adaptation may occur within the consumer’s domain, the provider’s domain, or can be included in the RLUS implementation, depending on agreements between trading partners and deployment context. See Section § 5.3 Semantic Profiles and Semantic Signifier for a discussion about semantic signifier structure.

---

13 An RLUS instance can support multiple Semantic Signifiers that can include textual and non-textual data.

14 As a Medication Statement R-MIM (PORX_IN040010UV) or Scheduling D-MIM (PRSC_DM000000UV). Starting from a D-MIM should be considered if a pre-existing messaging standard is too fine grained in respect of service scope.

15 This aspect is related to the service abstraction principle from technology, internal functionality, logic of implementations (see Thomas Erl, SOA Principles of Service Design, Prentice Hall, 2007). The RLUS service model is accordingly designed to avoid coupling between the exposed interface and internal implementation.
2. Implementation Considerations

These considerations identify the participant of the RLUS service architecture and simultaneously support the understanding of RLUS as neutral from a specific software architecture. All the deployment Scenarios presented here are supported by a common RLUS interface. The rationale is that RLUS makes no assumption about an implementation and is abstracted from the deployment architecture behind its service contract. Consequently, the deployment scenarios presented here are only illustrative and are not normative.

So, from the standpoint of this standard, any RLUS deployment meets these goals:

- It simplifies interoperability between organizations and systems, or departments, by means of a common and standardized interface independent from an underlying implementation
- It allows compliance to this standard to be testable and verifiable
- It is semantically extensible to meet expanding business needs
2.1 Deployment Scenarios

RLUS is explicitly an interface specification, not an implementation specification. As such, it is intended to be an interoperability mechanism among organizations. In addition, there is nothing inherent in the specification that precludes its use within a single organization, allowing a standardized method of record registry, location, and access. Conformance to the specification is asserted against profiles of the specification rather than against the specification itself.

Consequently all scenarios herein should be considered non-normative with regard to conformance to the RLUS standard. They are offered for explanatory purposes only and other topologies can be realized on the basis of the RLUS specification.

![Deployment Diagram](image)

**Figure 1: Representative deployment of RLUS within an organization (intra-organizational)**

RLUS can work in multiple different deployment topologies, and can be used to support different types of information. These are all deployment decisions and sensitive to deployment context, and are valid insofar as they are explicitly profiled.

RLUS exposes its standardized interface that abstracts the local implementation characteristics. So each RLUS consumer:

- is decoupled from the internal applications technology.
- is decoupled from the local application topology
2.1.1 Intra-organizational deployment

Figure 1 describes an exemplary intra-organizational deployment scenario where RLUS takes the role of an enterprise resource manager. In this scenario RLUS mediates among consumers and, as an example, a registry and one or more repositories.

2.1.2 Cross-organizational deployment: centralized

Figure 2 describes a possible cross-organizational topology where it works as an intermediary for other RLUS (or not) implementations. It could be a regional or a national metadata registry or it could be a service set up between 2 or more data sharing partners that composes local implementations. In Figure 2 the...
cross-organizational RLUS works as a full mediator and the consumers communicate only with the central RLUS instance that composes and collects information received from RLUS instances.

In general it’s relevant to stress the fact that RLUS can be used with different internal technical implementations within a registry and repository architecture or an integrated application based on, as an example, an XML database or a RIMBAA application.

### 2.1.3 Cross-organizational deployment: registry and repository architecture

A variant in this cross-organizational setting can be a cross organizational RLUS that works only as a registry. A consumer, after calling the RLUS central instance for location of resources, directly calls local RLUSs for retrieval of the resources previously localized.

---

**Figure 3:** Representative RLUS deployment in a registry and repository scenario

### 2.1.4 Cross-organizational deployment: peer to peer

Figure 4 represents another cross-organizational setting with a peer-to-peer architecture. In this topology no central RLUS instance exists and the different RLUS instances take the role of provider and consumer of...

---

16 HL7 RIM-Based Application Architecture.

17 This is similar to the IHE-XDS registry and repository architecture.
öyle bir topoloji. Bu durumda bir katılımcının arama isteğini gerçekleştirmek için ilgili bilgileri erişim yeteneğine sahip olması gerekir.) veya daha karmaşık bir peer-to-peer protokol kullanılabilir. 186

**Figure 4: Representative RLUS deployment in a cross-organizational p2p setting (inter-organizational)**

Görunüşte, bu tür ortamlarda RLUS'ın misyonu yerine getirmek için ekstra işlevlilik talep edilmemektedir._Deployman daha çok politik, anlaşmalar ve sahibi olma konularına çevrilir. Bu özellikle RLUS bir sistemdeki repository güncellemelerini yönetmek zorunda olduğu durumlarda daha net. 191

İnteraksiyonlar, hizmet sağlayıcısı ve hizmet tüketici arasında konsantredir ve farklı dağıtım ile fonksiyonel uyumluluk için açık conformance profilleri göre kullanıcı. 193

Semantic Signifiers, işlevsel profili ve destekleyen işsel durumları kastedilen en az şekilde RLUS uyumlu olarak tanımlamak. 195

Yeni birelerin eklenmesi, dışsal bileşenlerin derinlemesine bağlı Olmadığı bilgilendirici semantiğe bağlı olarak, 198
3. Business context

3.1 The reason why the service is necessary

Service orientation, in general, makes systems and architectures less brittle and more durable (e.g. less subject to change) mainly because they operate a separation of concerns between interface and implementation. Systems can evolve without undesirable coupling with other systems while ensuring an appropriate and stable level of functionality internally and/or externally to an organization.

RLUS, by providing functionality inherented from the resource location, retrieval and update service and working as an information broker, defines an essential component of interoperability. Further, by explicitly creating this component as a service, the need to adhere to the system architecture being used by the organization is reduced to aspects implied in the service interface definition, allowing more narrowly defined roles and functionality to be supported among organizations.

In the RLUS specification the term resource is intended as any intelligible and identifiable entity in a context, as a business object, a document and in general any identifiable set of information useful in a specific business context. For example, in the context of use of an RLUS interface as the interaction with a demographics service, a demographics record would be a resource.

This specification approaches this wide scope by creating a framework for measurable and testable levels of interoperability (see section § 5.4). These conformance levels allow for an RLUS compliant service to respond to different business partners according to different security models and functional profiles.

This flexibility in turn provides the basis for Service Level Agreements (SLAs) between sharing partners and organizations. Currently, SLAs are written in an ad-hoc manner and are derived from an arbitrary starting point. The RLUS model provides a clear starting point on which these agreements may be based.

3.2 Business Scenarios

3.2.1 Scenarios overview

All scenarios herein should be considered non-normative with regard to conformance to the RLUS standard. They are offered for explanatory purposes only.

However the scenarios are useful to identify and understand the RLUS capabilities without imposing any limit of alternative business usage of RLUS Service. Only the RLUS contract (see § 4. Detailed Functional Model) defines the limits of service usage.

However each scenario presented here is tied to one or more capabilities, and as such, is suggestive of exemplary functional profiles. Different Semantic Profiles can obviously meet these business scenarios, and in any case, again, should be considered illustrative for the purposes of demonstrating capabilities and scope.

---

18 See Glossary (Appendix A) for the definition of Information Broker term.

19 See Glossary (Appendix A) for the definition of Resource term.
These scenarios are not intended to preclude different deployed instances, localization, or extension. For example, some sort of tracing or auditing would likely be implemented to support versioning. As this concept speaks to a specific implementation condition, it is not included in the following scenarios.

We want to stress this point: the scenarios described here can imply that RLUS is essentially related to document management, but even though documents can certainly be a relevant usage of RLUS, by means of the Semantic Signifier concept, any type of informative resource can be managed.

The following scenarios are grouped to identify and understand different sets of capabilities:

- **Core Runtime Scenarios**

  These are the basic runtime scenarios of Retrieve, Locate, and Update Resource

- **Administrative Scenarios**

  These are more sophisticated Scenarios related to complex resource management capabilities such as linking and merging resources

- **Introspective Scenarios**

  These are the advanced scenarios related to Semantic Signifiers, Conformance Profile discovery and management. These scenarios are introspective in the sense that they are about the characteristic of an RLUS instance in itself.

3.2.2 **Scenario Actors**

The following actors take a role in the business scenarios:

- **Caregiver** – Caregivers are the main type of human RLUS beneficiary. Please consider that in general a RLUS beneficiary is not necessarily a caregiver. In different usage scenarios the beneficiary can be the patient themselves or, as an example, an epidemiologist or a scientist that has access to a, probably, anonymized resource. Moreover a RLUS beneficiary can be another system (not human), e.g. a Decision Support System\(^{20}\), an ETL system of Data warehouses or other type of applications.

- **Caregiver System** – Any healthcare information system component that interacts with the user (the caregiver) and that interacts with other systems components including an RLUS Instance, to fulfill user needs. If the RLUS beneficiary is a system this component is degraded to a simple service consumer.

- **RLUS Instances** – The Retrieve, Locate, and Update Service expose the RLUS capabilities in a standard RLUS interface and acts as a mediator for underlying systems.

---

• **Various RLUS Storage**—Contains assets, information, data, and other objects that are interesting to RLUS consumers. It is representative only of functionality without a normative restriction on the nature of information and architectural choice within an organization. RLUS storage can be Registries and/or Repositories realized with various technologies (see § 2. Implementation Considerations) or an integrated application (a XML Database-based application, a complex EHR implementation or a RIMBA application, etc).

### 2673.3 Core Runtime Scenarios

This set of scenarios represents the “core” of RLUS basic usage at runtime.

### 2693.3.1 Locate Resources

#### 2703.3.1.1 Scenario

A caregiver has a new patient and needs to see information about his/her history. He/She accesses a list of all resources that are available for this patient within the local domain. Two separated local organizational registries inform the local RLUS implementation. RLUS compiles a list of available resources and makes it available to the caregiver. The caregiver can save the list for future analysis or can continue retrieving one or more resources (see § 3.3.2 Get Resource scenario).

#### 2763.3.1.2 Interfaces Used

- Locate Resources (see § 4.4.1)

#### 2783.3.1.3 Scenario Details

![Diagram](image-url)
3.3.2 Get Resource

3.3.2.1 Scenario

A caregiver has the list of the available resources previously recovered in Locate scenario or can search a resource that can have only one instance (e.g. a patient summary). Upon selection from the list or by an appropriate query he or she can retrieve the requested resource.

3.3.2.2 Interfaces Used

- Get Resource (see § 4.4.2)

3.3.2.3 Scenario Details

Figure 5: Locate Resource scenario

Figure 6: Get Resource Scenario
3.3.3  List and Get Resources

3.3.3.1  Scenario

A caregiver (e.g. a Pharmacist or a GP) wants to recover a recent prescription for a patient. In this scenario the Locate plus Get functionality is not useful because a selection from a list makes no sense. The relevant information is in the content and not in the metadata of the resource. So the caregiver requests the resource type directly by parameters (e.g. prescription document type, data range or state) and directly retrieves the document set.

3.3.3.2  Interfaces Used

- List And Get Resources (see § 4.4.3)

3.3.3.3  Scenario Details

Figure 7: List and Get Resource Scenario
3.3.4 Put Resource

3.3.4.1 Scenario

3.3.4.2 Interfaces Used

3.3.4.3 Scenario Details

Figure 8: Put Resource Scenario
3.3.5 Initialize Resource

3.3.5.1 Scenario

A caregiver (e.g. a Pharmacist or a GP) wants to publish a resource (e.g. a prescription) for a patient. The specific deployment scenario is that the resources are stored and exposed by RLUS compliant storage but it’s necessary to notify the resource creation event to a cross-organizational RLUS instance. The caregiver writes the resource and forwards it to an RLUS local instance. The RLUS local instance stores the resource and initializes the resource by calling a cross-organizational RLUS instance that exposes it by pointing the resource at the local RLUS repository for future retrieval.

Note: a variation in this scenario is that the responsibility of calling the cross-organizational RLUS instance is taken directly by the caregiver system. However it’s interesting to note that the scenario presented here hides the complexity of the architecture from the caregiver system.

3.3.5.2 Interfaces Used

- Put Resource (see § 4.4.4)
- Initialize Resource (see § 4.4.5)

3.3.5.3 Interaction Details

![Diagram of Initialize Resource Scenario]

Figure 9: Initialize Resource Scenario
3.3.6 Discard Resource

3.3.6.1 Scenario

A caregiver (e.g. a GP) has discovered that a resource, previously published, contains errors and decides to discard it from a RLUS instance. The caregiver calls RLUS to delete the resource. The resource is logically discarded from the RLUS instance.

Note: In general this is likely to be a logical deletion, due to the nature of healthcare data, and not a physical deletion in most practical implementations. However it’s a question of internal policy of an organization or domain.

3.3.6.2 Interfaces Used

- Discard Resource (see § 4.4.6)

3.3.6.3 Scenario Details

![Discard Resource Scenario Diagram]

Figure 10: Discard Resource Scenario
3.4 Administrative Scenarios

These scenarios are related to more complex resource management capabilities such as linking and merging resource.

3.4.1 Link Resources

3.4.1.1 Scenario

A pharmacist lists and retrieves the last orders for a patient. After dispensing, the pharmacist puts a resource that describes the dispense action and dispensed product. After this the pharmacist creates a link between an order and the dispense document.

3.4.1.2 Interfaces Used

- List and Get Resources (see § 4.4.3)
- Put resource (see § 4.4.4)
- Link resources (see: § 4.5.1)

3.4.1.3 Scenario Details

Figure 11: Link Resource Scenario
3.4.2 Unlink Resources

3.4.2.1 Scenario

A specialist has previously linked an order with the report document that he/she has produced and published. Immediately after this he or she discovers that the link is wrong because it is not the correct order. So he or she unlinks the documents and can link the correct documents.

3.4.2.2 Interfaces Used

- List and Get Resources (see § 4.4.3)
- Put Resource (see § 4.4.4)
- Link Resources (see: § 4.5.1)
- Unlink Resources (see § 4.5.2)

3.4.3 Merge Resources

3.4.3.1 Scenario

An order from a Specialist for a set of exams has generated two laboratory reports. The specialist wants to merge the two reports into one document to simplify the reference by a referral document. The RLUS
374implementation is informed of the request and merges the two laboratory reports. After merging the 375specialist completes and puts its report document with a reference to the merged laboratory reports.

376### 3.4.3.2 Interfaces Used

377- List and Get Resources (see § 4.4.3)
378- Merge Resources (see § 4.5.3)

379### 3.4.3.3 Scenario Details

380![Diagram of Merge Resource Scenario](image)

381**Figure 13: Merge Resource Scenario**

382### 3.4.4 Unmerge Resources

383#### 3.4.4.1 Scenario

384A patient is discharged from an inpatient facility. A caregiver finds that two discharge summaries are 385exposed in the RLUS and decides to combine them into a merged document to better describe the patient 386stay. RLUS receives the request and merges the two documents. Some days later the caregiver decides that 387only one of the original documents are truly relevant and that document should be linked with another 388new document. So the RLUS implementation is informed and unmerges the documents. After this the
389 caregiver, with the help of the RLUS implementation, can link a new document with one of the original discharge summaries.

391 3.4.4.2 Interfaces Used

392 - Merge Resources (see § 4.5.3)
393 - Unmerge Resources (see § 4.5.4)
394 - Link Resources (see § 4.5.1)

395 3.4.4.3 Scenario Details

396 Figure 14: Unmerge Resource Scenario

398 3.5 Introspective Scenarios

399 3.5.1 List and Describe Semantic Signifiers and Conformance Profiles

400 3.5.1.1 Scenario

401 A caregiver requests information about a new patient. His/her advanced EHR system queries the appropriate RLUS instance for each service's self-description interface and discovers that the external organization supports CDA R2 as well as Continuity of Care Documents (CCD) for their retrieval interface.

402 The EHR is now aware of the semantics and operation of the external system, and may create an automated transformation or notify a system analyst that further interaction is necessary or continue with locate, get or list resource if the semantic signifiers are directly compatible.
3.5.1.2 Interfaces Used

- List Semantic Signifiers (see § 4.6.2)
- Describe Semantic Signifier (see § 4.6.3)
- List Conformance Profiles (see § 4.6.1)

3.5.1.3 Scenario Details

Figure 15: List and Describe Semantic Signifiers Scenario

3.5.2 Put Semantic Signifier

3.5.2.1 Scenario

A new Semantic Signifier definition (a new kind of resource) must be created in a RLUS instance. A technical expert, supported by caregivers, creates the new definition or uses a profiled definition previously created (e.g. a CDAr2 template). The new Semantic Signifier is submitted to the RLUS instance. The RLUS instance stores the new Semantic Signifier and prepares the RLUS instance to manage the new kind of resource.

3.5.2.2 Interfaces Used

- Put Semantic Signifier (see § 4.6.4)
3.5.2.3 Scenario Details

Figure 16: Put Semantic Signifier Scenario
4. Detailed Functional Model

4.1 Structure of the Service

Starting from the business scenario we can identify the set of RLUS capabilities. Table 1 defines a matrix that maps the business collaboration messages and RLUS capabilities.

Table 1 – Business Message and Capabilities

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Discard Resource Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discard Resource Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get Resource Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Get Resource Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialize Resource Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialize Resource Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link Resources Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link Resources Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Describe Semantic Signifier Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Describe Semantic Signifier Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Conformance Profiles Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Conformance Profiles Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Semantic Signifiers Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Semantic Signifiers Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List and Get Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List and Get Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Locate Resources Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Locate Resources Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Merge Resources Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Merge Resources Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Put Resource Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Put Resource Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Put Semantic Signifier Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Put Semantic Signifier Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unlink Resources Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unlink Resources Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In order to provide the maximum implementation flexibility, this functional model defines several capability groups for RLUS. These groups identify a subset of the RLUS functions as pertinent to a specific logical context.

The grouping, as a whole, defines a set of basic Functional Profiles for RLUS. Functional Profiles are intended as an aggregation of capabilities that are supported by a particular instance of RLUS:

- **RLUS Runtime** – The capabilities contained in this group define the core RLUS “CRUD” functionality as retrieve, locate, update and discard.

- **RLUS Administrative** – This group of capabilities allows consumers to operate more sophisticated functions such as to link and unlink between resources, to merge and unmerge resources.
• **RLUS Introspective**\(^{21}\) – This capabilities group includes mainly introspective functions related to Semantic Signifier management such as mechanisms to describe a Semantic Signifier, to list Semantic Signifiers, to put a Semantic Signifier and the capability to list supported Conformance Profiles.

\(^{21}\) See Glossary (Appendix A) for definition *Introspective* term.
4.2 RLUS Conformance

RLUS explicitly makes no distinctions at a functional level regarding semantics of the underlying systems. Instead, it provides for a semantic profile as part of RLUS conformance profiles. This allows definition, publication, and discovery of vital semantic artifacts among partners through RLUS interfaces without requiring strict, tightly coupled integration. Thus, RLUS does not preclude a semantic interoperability strategy to be realized, though it would likely depend on other factors (for example, a common terminology service and / or a transformation service). This improves RLUS as an interoperability mechanism by relegating the issue of semantic interoperability to the trading partners, allowing semantic transformations to be performed at the least cost for the most derived value.

RLUS provides appropriate mechanisms to manage this flexibility by means of Functional and Semantic profiles.

---

22 For the purposes of this specification, the terms compliance and conformance are based upon definitions from the HL7 Services-Aware Interoperability Framework (SAIF).
• **Functional Profiles** are the aggregation of all operations that are supported by a particular instance of RLUS. An RLUS instance may support one or more defined Functional Profiles Levels or can support a specific narrowed and business-lead set of capabilities (see § 5.2 Functional Profiles).

• **Semantic Profiles** are the aggregation of all Semantic Signifiers that collectively describe the semantics of the interactions supported by a particular RLUS instance (see § 5.3 Semantic Profiles and Semantic Signifier).

Functional Profiles together with Semantic Profiles define the **Conformance Profile** of a specific RLUS instance. A Conformance Profile is directly derived from the agreements implemented with a business partner. So a single RLUS instance can be tailored to respond to different real-world business partners depending on the underlying agreements and needs. Conformance Profiles are discussed in Section § 5.4 Conformance Profiles.

### 4.3 Assumptions and Dependencies

RLUS is autonomous, not dependent on other kinds of services and is decoupled from specific implementation details. Consequently no reference to internal technical characteristics exists in a conformant RLUS implementation. It is recognized that RLUS is a specific SOA service, and as such is intended to complement other SOA services as might exist within an architecture. For example, RLUS does not take into account security considerations, as those capabilities would be provided by other services.

#### 4.3.1 Assumptions

1. **Resource Identification and Location**

RLUS explicitly requires that each resource or asset that is going to be exposed via RLUS should be uniquely and explicitly identified and located either explicitly or implicitly. If this is not explicitly part of the semantic signifiers used to represent the data, then these two information components will need to be available to RLUS implicitly.

For example, RLUS could be used to expose Health Record Summaries available as discrete documents that have a physical location and an identifier. RLUS could also be used to retrieve laboratory orders from a database, or from a composed set of services. In this case, the laboratory orders would not have a physical location per se, nor an identifier per se. Both instances of RLUS are valid depending on the semantic and functional profiles specified.

2. **Entity Identification**

Entities identified within semantic structures, such as patients, must have unique identifiers within a particular deployment context (for example, IHE’s concept of Affinity Domains). For instance, Mrs. Jones is almost certainly uniquely identifiable in Organization A and in Organization B. Organization B needs to be able to uniquely identify Mrs. Jones as the organization’s members access A’s systems. This level of entity identification is supported by HSSP’s Identity Cross-Reference Service (IXS), though RLUS does not assume how this functionality is provided.
4.3.1.3 Information Quality

RLUS implicitly occupies a role of being the external gateway to authentic, authoritative data within an organization. However, RLUS can only guarantee the quality of the structure of the information rather than the information itself, for which it relies on the underlying systems. Put another way, RLUS rigorously adheres to functional and semantic profiles, but can only deliver information as well as the systems that it represents.

4.3.2 Dependencies

4.3.2.1 Dependencies on other HSSP Infrastructure Services

RLUS service, as a SOA service, has no formal dependence on other services. However, RLUS implementations can be greatly strengthened through the availability of strong support a standardized service inventory in a clean service oriented architecture. Standards such as the Common Terminology Service R2 (CTS2), the Clinical Decision Support Service (CDSS), the cited Identity Cross-Reference Service (IXS) and the HSSP service inventory members in general, form a foundational services ecosystem well integrated with RLUS.

Despite these benefits, dependencies on external components are intentionally loose to provide significant flexibility in use and in deployment. Organizations may or may not build architectures on these standard infrastructure pieces. As a result, these dependencies are instead identified as pre-conditions throughout the specification. So Organizations may choose to omit these capabilities, or use internal or legacy alternatives so long as the specified preconditions are somehow addressed.

4.3.2.2 Cross-organizational RLUS and Discovery and Description Services

A cross-organizational implementation of RLUS (centralized or peer-to-peer) is not another profile to be described rather it is only a special deployment case. There are no functional differences between an RLUS implemented locally and a regional implementation that spans organizations. What is behind an RLUS interface is not of interest to the consumer and is hidden.

Each implementation may operate at any one of the interface conformance levels, each may support the extended use cases, and so on. For example, it will still need some sort of entity identification service, but it will need one that spans organizations. Similarly, its security profile will need to interact with different systems in different ways under different agreements.

4.3.2.3 Support for Semantic Signifiers

RLUS requires the information that it exposes to conform to some structures that can be used for classification and organization. The term chosen for these structures is Semantic Signifier, and they are essential elements of the RLUS interface.

Semantic Signifiers provide common information “building blocks” that RLUS interfaces use to convey content. Because RLUS exists at the service level, and because of the requirement for services to be both composable and self-contained, RLUS enables information, that it exposes, to be described in terms of these signifiers through the functional interfaces (e.g., there is a mechanism in the interface for the dynamic discovery of the information content and profiles supported by an RLUS implementation). RLUS supports rich semantics in both the request and response portions of its interface lifecycle.
For the purposes of RLUS, Semantic Signifiers have certain characteristics that allow simple interfaces to support rich semantic interactions:

- They are expressible in a format so that consumers of an RLUS service can discover the notations required to access certain functional components
- They are described by reference or by value through the RLUS interfaces so as to be verifiable
- Finally, these structures are traceable to a particular semantic profile so as to be tested for conformance at either run time or design time.

For example, if patient data is intended to be searched by a patient identifier, then that identifier needs to be included in the semantic representation of that data so that it may be queried. While RLUS makes only minimal assertions regarding Semantic Signifiers, it is inferred that these same Semantic Signifiers will support the profiled functionality of RLUS. Semantic Signifiers are described in § 5.3 Semantic Profiles and Semantic Signifier.

4.4 Basic Run-Time Capabilities

4.4.1 Locate Resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Given a set of parameters, returns a list of matching resource locations and a set of metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precondition</td>
<td>• The consumer has access to an appropriate query structure before making the request</td>
</tr>
<tr>
<td>Inputs</td>
<td>• Semantic signifier identifying the information model that is used to express the query parameters</td>
</tr>
<tr>
<td>Or</td>
<td>• Query parameters, expressed in conformance with the semantic signifier</td>
</tr>
<tr>
<td>Or</td>
<td>• Query by example, expressed in conformance with the semantic signifier</td>
</tr>
<tr>
<td>Outputs</td>
<td>• Location of resources (RLUS instance endpoint where the resources are located) and id of the resources</td>
</tr>
<tr>
<td>Or</td>
<td>• Metadata sets for the resources</td>
</tr>
<tr>
<td>Post-conditions</td>
<td>• A list of resources parameterized by the inputs is available to the consumer</td>
</tr>
<tr>
<td>Exception</td>
<td>• The RLUS instance is unavailable or unable to provide the service</td>
</tr>
</tbody>
</table>

See Glossary for the definition of Query By Example term.
Conditions
- Resources not found

Notes
The intent of this interface is to return lists of resources that match a set of search criteria located in the local RLUS instance and in other RLUS instances. The operation doesn’t change the resources states and has no side effects (locking, etc.)

5504.4.2 Get Resource

Description
Given a set of parameters that uniquely describes a resource, returns the identified resource.

Precondition
- Resource Exists
- Resource is available, is not discarded, is not corrupt

Inputs
- Unique Resource Identifier
  Or
  - Semantic signifier identifying the information model that is used to express the query parameters
  - Query parameters, expressed in conformance with the semantic signifier
  Or
  - Query by example\(^\text{24}\), expressed in conformance with semantic signifier

Outputs
- The resource

Post-conditions
- The resource is available to the consumer’s system

Exception Conditions
- The RLUS instance is unavailable or unable to provide the service
- Resource not found
- More than one resource found

Notes
This capability returns a single resource, if a query is used and more than one resource matches, RLUS instance will return an appropriate error condition.

\(^{24}\) See Glossary for the definition of Query By Example term.
### 4.4.3 List and Get Resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Given a set of parameters, returns a set of resources that matches the parameters</th>
</tr>
</thead>
</table>
| Precondition | • At least one Resource Exists  
• Resource is available, is not discarded, is not corrupt |
| Inputs | • Semantic signifier identifies the information model that is used to express the query parameters  
• Query parameters, expressed in conformance with the semantic signifier  
Or  
• Query by example\(^{25}\), expressed in conformance with semantic signifier |
| Outputs | • The resources set |
| Post-conditions | • The resources are available to the consumer’s system |
| Exception Conditions | • The RLUS instance is unavailable or unable to provide the service  
• Resources not found |
| Notes | This capability is a combined query plus retrieve function and returns a set of resources. |

### 4.4.4 Put Resource

<table>
<thead>
<tr>
<th>Description</th>
<th>Stores a resource in underlying RLUS storage and makes it accessible.</th>
</tr>
</thead>
</table>
| Precondition | • There is a need to expose a particular resource through RLUS.  
• Semantic Signifier describing the resource is compatible with the RLUS Instance |
| Inputs | • Resource  
• Semantic Signifier |
| Outputs | • Success / Failure Notification |
| Post-conditions | • Resource is accessible by RLUS interface |

\(^{25}\) See Glossary for the definition of Query By Example term.
### Initialize Resource

<table>
<thead>
<tr>
<th>Description</th>
<th>The consumer notifies a resource creation but the resource is not necessarily stored in the underlying RLUS instance system. Other initialization contexts can be possible on an instance by instance basis.</th>
</tr>
</thead>
</table>
| Precondition | • There is a need to expose a particular resource through RLUS in a specific RLUS network.  
• Semantic Signifier describing the resource is compatible with the RLUS Instance involved |
| Inputs      | • Initialization context  
• Resource  
• Semantic Signifier |
| Outputs     | • Success / Failure Notification |
| Post-conditions | • Resource is accessible by RLUS interface  
• If appropriate, the underlying repository’s registry is updated |
| Exception Conditions | • The RLUS instance is unavailable or unable to provide the service  
• The underlying storage is unavailable  
• The Semantic Signifier of the resource is not supported by this implementation of RLUS |
| Notes | This operation is similar to Put, but the resource is not necessarily stored in underlying storage. Initialize operation can be used to notify a new resource in a RLUS network or in other various scenario specified by an initialization context. |
### 554.4.6 Discard Resource

<table>
<thead>
<tr>
<th>Description</th>
<th>Removes a resource from a RLUS instance (in the underlying storage), or makes it unavailable. Also removes the RLUS reference to that resource (e.g. link) or makes it unavailable.</th>
</tr>
</thead>
</table>
| Precondition | - The resource exists  
- The resource is available  
- There is no longer a need to make this resource available via RLUS  
- There is no longer a need for the resource |
| Inputs |  
- Unique Resource ID  
Or  
- Semantic signifier identifies the information model that is used to express the query parameters  
- Query parameters, expressed in conformance with the semantic signifier  
Or  
- Query by example\(^{26}\), expressed in conformance with semantic signifier |
| Outputs | Success / Failure notification |
| Post-conditions | The resource has been made unavailable, and is no longer accessible. |
| Exception Conditions | - The RLUS instance is unavailable or unable to provide the service  
- The resource is unavailable, is in use, or is locked |
| Notes | In general a discard should be only a logical (not physical) deletion of a resource. It is likely that, in a RLUS implementation, some sort of trace should be in effect to see what is being removed, by whom, why, etc. |

---

\(^{26}\) See Glossary for the definition of *Query By Example* term.
### Preconditions

- The resources exist
- The resources are available

### Inputs

- Source Unique Resource ID
- Target Unique Resource ID
- Link Type

### Outputs

- Success / Failure notification

### Post-conditions

- The resources are linked and the appropriate metadata are updated or created

### Exception Conditions

- The RLUS instance is unavailable or unable to provide the service
- A linked resource is unavailable or does not exist

### Notes

The link can be contextualized via a Link Type parameter. This parameter can be a reference on the element of the Semantic Signifier structure involved in the link operation.

The Link operation result, generally, should be traced to an appropriate element specified in the Semantic Signifier.

---

#### 5.5.2 Unlink Resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Discard a relation between two resources of a RLUS instance</th>
</tr>
</thead>
</table>

### Preconditions

- The resources exist
- The resources are available
- The link exist

### Inputs

- Source Unique Resource ID
- Target Unique Resource ID
- Link Type

### Outputs

- Success / Failure notification

### Post-conditions

- The resources are unlinked and the appropriate metadata are updated

### Exception Conditions

- The RLUS instance is unavailable or unable to provide the service
- A linked resource is unavailable or does not exist
- The link does not exist
Notes

It is likely that, in a RLUS implementation, some sort of trace should be in effect to see what is being removed, by whom, why, etc.

The Unlink operation result, generally, should be traced to an appropriate element specified in the Semantic Signifier.

### 5594.5.3 Merge Resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Merge two resources of a RLUS instance. The operation transforms two existing resources in a combined resource and removes (discards) the previous resources</th>
</tr>
</thead>
</table>
| Precondition | • The resources exist  
• The resources are available |
| Inputs       | • Source Unique Resource ID  
• Target Unique Resource ID |
| Outputs      | • Success / Failure notification  
• If successful merge, the operation returns the Unique Resource ID of the new combined resource |
| Post-conditions | • The resources are merged |
| Exception Conditions | • The RLUS instance is unavailable or unable to provide the service  
• A merged resource is unavailable or does not exist |
| Notes | The merge is a sophisticated operation and the successful completion is in dependent on the capabilities of the underlying business logic.  
An appropriate error must be set if the merge fails.  
It's also necessary that an appropriate compensation mechanism is in effect for failed merges and also for the companion unmerge operation |

### 5604.5.4 Unmerge Resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Unmerge two resources of a RLUS instance. The operation discard the merged resources and exposes the original resources (as-is before the previous merge operation)</th>
</tr>
</thead>
</table>
| Precondition | • The resource exists  
• The resource is available |
The resource is a result of a previous merge operation

### Inputs
- Source Unique Resource ID
  - Or
  - Semantic signifier identifies the information model that is used to express the query parameters

  - Query parameters, expressed in conformance with the semantic signifier

### Outputs
- Success / Failure notification
- The operation returns the Unique Resource IDs of the original resources (before a merge operation)

### Post-conditions
- The resource is unmerged

### Exception Conditions
- The RLUS instance is unavailable or unable to provide the service
- The resource is unavailable or does not exist
- The resource is not a result of a previous merge operation

### Notes
The unmerge capability is dependent on the capability of underlying business logic and storage (see also merge operation notes).

#### 4.6 Introspective Capabilities

#### 4.6.1 List Conformance Profiles

<table>
<thead>
<tr>
<th>Description</th>
<th>Produces a list of conformance profiles supported by this RLUS instance</th>
</tr>
</thead>
</table>
| Precondition | • Consumer has agreed to exchange data with the organization via RLUS  
               • RLUS instance knows which conformance profiles that it supports (e.g., set by configuration) |

| Inputs | None |

| Outputs | List of conformance profiles, including identifier, version, registering authority, and description |

| Post-conditions | A list of the supported conformance profiles has been delivered to the consumer |

| Exception Conditions | The RLUS instance is unavailable or unable to provide the service |
• Conformance profiles are not available

Notes

As particular instances of RLUS will make assertions regarding conformance, it is desirable to allow RLUS to self-describe its conformance level at design-time and run-time. This description in turn allows the conformance assertion to be tested.

A code system will likely be needed, either globally or within each context / domain, which identifies conformance profiles

5634.6.2 List Semantic Signifiers

<table>
<thead>
<tr>
<th>Description</th>
<th>Lists semantic signifiers that are available from this RLUS implementation, for example, what is available and how it is retrieved</th>
</tr>
</thead>
</table>
| Precondition | • Consumer has agreed to exchange data with the organization via RLUS  
• Semantic signifiers are describable and available  
• Semantic signifier is able to be modeled logically |
| Inputs | • None |
| Outputs | • List of semantic signifiers that are available for the entire RLUS implementation, including:  
  ➢ Semantic signifiers that specify the format and semantics of resources that can be returned by the RLUS (resource semantic signifiers)  
  ➢ For each of the resource semantic signifiers, zero or more semantic signifiers that specify the information models that can be used to query for the data of interest (query parameter semantic signifiers) |
| Post-conditions | A list of semantic signifiers and their descriptions has been delivered to the consumer |
| Exception Conditions | • The RLUS instance is unavailable or unable to provide the service  
• List of semantic signifiers are not available |
| Notes | Having an available list and description of each semantic signifier used by an organization is a minimum requirement for all levels of RLUS conformance.  
It is desirable to do more than simply list the signifiers. For example, each signifier may point to a schema, or to an explanation, or to an
4.6.3 **Describe Semantic Signifier**

<table>
<thead>
<tr>
<th>Description</th>
<th>Retrieves a description and formal model of a local semantic signifier.</th>
</tr>
</thead>
</table>
| **Precondition** | • Submitter has agreed to exchange data via RLUS with the organization  
• Semantic signifiers are describable and available  
• Semantic signifier is able to be modeled logically |
| **Inputs** | • Unique Semantic Signifier Identification |
| **Outputs** | • The description of the semantic signifier  
• The formal model expressing the semantic signifier as expressed in a Localized Information Model (LIM).  
• The Status of the semantic signifier |
| **Post-conditions** | • The model of the semantic signifier is available for further operations |
| **Exception Conditions** | • The RLUS instance is unavailable or unable to provide the service  
• Semantic Signifiers are unavailable  
• Formal Model is missing or unreachable  
• Semantic signifier has been marked as invalid |
| **Notes** | This is an aspect of the functional model that is a key to interoperability. There needs to be an automated way of accessing the formal models that describe an organization’s content so that information can flow. See Section § 5.3 for a discussion of semantic signifiers. |

5.6.4.6.4 **Put Semantic Signifier**

<table>
<thead>
<tr>
<th>Description</th>
<th>Send a new or updated Semantic Signifier to a RLUS instance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Precondition</strong></td>
<td>• There is a need to expose a new or updated Semantic Signifier.</td>
</tr>
<tr>
<td><strong>Inputs</strong></td>
<td>• Semantic Signifier</td>
</tr>
</tbody>
</table>
| **Outputs** | • Semantic Signifier is automatically implemented or stored (in this
<table>
<thead>
<tr>
<th>Post-conditions</th>
<th>The Semantic Signifier is accessible in RLUS interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exception</td>
<td>The RLUS instance is unavailable or unable to provide the service</td>
</tr>
<tr>
<td></td>
<td>The Semantic Signifier cannot be automatically implemented</td>
</tr>
<tr>
<td>Notes</td>
<td>As the implementation of the Put operation is complex, it is recommended that careful review be conducted before this result is exposed via the interface.</td>
</tr>
</tbody>
</table>

567 **Profiles**

568 **5.1 Introduction**

569 Due to the fact that RLUS represents generic functional capabilities, specific profiles should be defined in an implementation or in a set of implementations on an RLUS network. A profile describes the specific functions, semantic information and the overall conformance exposed by a RLUS implementation.

570 Briefly, the meaning of each of these types of profile is as follows:

571 - **Functional Profile:** a named list of a subset of the operations defined within this specification that must be supported in order to claim conformance to the profile.

572 - **Semantic Profile:** each semantic profile is defined by a Semantic Signifier specification. A Semantic profile identifies a named set of information description that is supported by an RLUS instance.

573 - **Conformance Profile:** this is a combination of a set of functional and semantic profiles and taken together give a complete coherent set of capabilities against which conformance can be claimed.

574 For example, a RLUS instance could support only a retrieval (get) and put capabilities and another only the location and/or list of resources (this is similar to IHE’s XDS) based on CDAr2 or a CCD Semantic Signifier. Additionally, profiles are additive, and may be present in multiplicity. However, any instance of RLUS must support at least one Conformance Profile.

575 By design, RLUS interfaces are loosely-coupled with the information qualities that they expose, though this support is enabled through the mandatory interfaces within profiles. Semantic profiles, like Functional Profiles, may be present in multiplicity.

576 No specific normative conformance profiles have been defined in this standard. Therefore, there are no normative functional or semantic profiles. The creation of conformance profiles is detailed in the HSSP’s Service Specification Development Framework guide and will be part of emerging SAIF. As a

---

27 For more information on this guideline and related materials, see the Infrastructure sub-group at http://www.healthinterop.org

28 HL7 Service Aware Interoperability Framework (see: http://en.wikipedia.org/wiki/HL7_Services_Aware_Interoperability_Framework)
589Consequence profiles are not normative except within the context of a conformance profile in a 590deployment context.

591It's expected that some profiles will be standardized by HL7 itself or by a profiling consortium such as IHE, 592however additional profiles may be created reflecting specific local or national requirements.

5935.2 Functional Profiles
594The functional compliance of an RLUS implementation specification is defined by a set of functional 595capabilities that are implemented and exposed. Each implementation specification is classified by its 596functional level of compliance:

597**Level 0 (L0).** This compliance level is defined by a subset of RLUS capabilities tailored for a specific 598interoperability scenario. As an example in a gateway scenario, a functional profile can be composed by List 599and Retrieve Resource, Put Resource and Discard Resource.

600**Level 1 (L1).** This level is composed of the Basic RLUS Run-Time capabilities fully implemented.

601**Level 2 (L2).** This level extends the L1 functional profile with the Administrative capabilities or the 602Introspective capabilities.

603**Level 3 (L3).** This level represents the complete RLUS implementation with the Basic RLUS Run-Time 604capabilities, the Administrative capabilities and the Introspective capabilities fully implemented.

605Note that these levels of compliance are useful to classify the standard RLUS implementation 606specification. A real world implementation can deactivate some capabilities in response to specific 607business needs and frequently can be expected to meet Level 0.

### Table 2: Conformance Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Capabilities group</th>
<th>Included Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>L0</strong></td>
<td>Tailored set of standard RLUS capabilities</td>
<td>Subset of standard capabilities tailored as needed</td>
</tr>
<tr>
<td><strong>L1</strong></td>
<td>Basic Run-Time capabilities</td>
<td>Run-Time capabilities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Locate Resources</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Get Resource</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• List and Retrieve resources</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Put Resource</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Initialize Resource</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Discard Resource</td>
</tr>
<tr>
<td><strong>L2</strong></td>
<td>Basic Run-Time Administrative capabilities</td>
<td>OR Introspective capabilities</td>
</tr>
</tbody>
</table>
### 5.3 Semantic Profiles and Semantic Signifier

610A Semantic Profile is a specific Semantic Signifier that can be implemented in a RLUS instance. This specification illustrates some exemplary semantic profiles in Appendix B.

611These illustrative profiles are not intended to meet every situation, business need, or implementation. They form the basis to establish relationships among trading partners, and identify the bare minimum functionality that can help RLUS to meet its intended interoperability mission.

612Thus, the profiles are provided for organizations creating their own RLUS interfaces, for vendors including RLUS implementations in their products, and other standards that rely on RLUS to meet certain needs in fulfilling their own conformance criteria.

613Further, conformance to each functional and semantic profile (e.g. an HSSP RLUS HL7 CDAr2 L0) is testable. This helps to insure a degree of interoperability and functionality across all implementations.

614The concrete implementation mechanism of Semantic Signifier, and consequently of a Semantic Profile, is not described here. The description of platform specific concrete implementation mechanisms are left to Service Technical Models.

615Semantic Signifiers express information that is inherent in several portions of a typical RLUS interaction. They can serve as a return type in that they classify and describe the information that is returned through the RLUS interface. They may also be used to classify and describe input parameters for RLUS functionality.

616Where the content being exposed through RLUS is HL7 Domain content, RLUS normatively requires that a semantic signifier be formally described by a Localized Information Model (LIM). LIMs provide a consistent means of allowing complex informational semantics to be communicated through service interfaces using industry standards. They are useful because they can allow the service consumer to associate knowledge of the instance over and above the inherent semantics of the information in the instance.

---

Figure 20: Abstract Structure of a Semantic Signifier

Figure 20 represents a logical view of components of a Semantic Signifier. A Semantic Signifier includes, at minimum, the schema of the resource (LIM) and can be completed with a data integrity ruleset, rendering rules and the searchable fragment of the resource. The Semantic Signifier definition can include the support of textual and non-textual data.

Multiple semantic signifiers can be supported in an RLUS instance without any explicit hierarchies. However, implicit hierarchies can be supported by specific implementations. As an example, an implementation can support a CDAr2 Semantic Signifier for Orders, a CCD Semantic Signifier for Record Summaries, as well as more generic, abstract Semantic Signifiers for generalized queries.

5.4 Conformance Profiles

The capabilities defined within this Service Functional Model and the Semantic Profiles could be attributed to different “Conformance Profiles.” The purpose of Conformance Profiles is to group together functions and Semantic Signifiers to form cohesive levels of operational capability against which implementations can be tested for conformance. Thus, interoperability between RLUS implementations is assured within a conformance profile. In other words, two RLUS implementations that conform to a specific Conformance Profile will be able to interoperate using the functions and the semantics described in that profile.

Figure 21 represents the abstract structure of a Conformance Profile. It is composed of a declaration of the compliance level of functional profiles, the detailed enumeration of operations allowed and the semantic signifiers supported by the profile.

---

30 By query mechanisms

31 The searchable fragment can define the subset of the resource that can be used in the query mechanism.
These profiles serve to support implementation variation while still promoting interoperability. Service Level Agreements made among organizations are then testable because they are informed by these profiles. Governance of these agreements is less ambiguous and more enforceable due to precise functional and semantic levels of interoperability that may be expected.
## Appendix A: Glossary

658 Citation of terms specific to this functional specification and not included in the overall HSSP Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset</td>
<td>An information resource that an organization chooses to make available through RLUS.</td>
</tr>
<tr>
<td>Caregiver</td>
<td>Refers to any person, group of people, or organization that imparts health care services to living subjects. This term is intended to be neutral and interpreted as reasonable and appropriate in any given implementation realm. Realm-specific interpretation examples include, but are not limited to: In the United States, “caregiver” in the context of this specification would commonly be interpreted as a physician or other licensed medical practitioner. In a broader interpretation of this specification, “caregiver” could also include a wide range of ancillary health care personnel such as physicians assistants, nurses, pharmacists, radiology technicians, medical educators, and dietitians, to name a few.</td>
</tr>
<tr>
<td>Component</td>
<td>As in Service Component. A component is a modular unit with well-defined interfaces that is replaceable within its environment. A component can always be considered an autonomous unit within a system or subsystem.</td>
</tr>
<tr>
<td>Consumer</td>
<td>A consumer requests a service exposed by a provider who then uses its capabilities to fulfill the service request and ultimately delivers value to the consumer. The interaction between a provider and a consumer is governed by a Service Contract where both parties are (directly or indirectly) bound by that contract. (From OMG, Service oriented architecture Modeling Language (SoaML) - Specification for the UML Profile and Metamodel for Services (UPMS), 1.0)</td>
</tr>
<tr>
<td>Cross-Domain RLUS</td>
<td>See Cross-Organizational RLUS. These are equivalent for the purposes of this specification.</td>
</tr>
<tr>
<td>Cross-Organizational RLUS</td>
<td>An inter-organization implementation of RLUS. A cross-organizational RLUS has a child relationship with several parent organizations. For example, it may be informed by one or more local RLUS implementations, utilize security implementations that are regional in nature, normalize entities (patients, providers) across organizations, and use prefixes to define resource identities at each parent organization. However a cross-organizational scenario can be realized with different architectures that do not involve a cross-organizational RLUS instance (see § 2.1.2, § 2.1.3, § 2.1.4)</td>
</tr>
<tr>
<td>Direct Information Broker</td>
<td>See Information Broker. A direct information broker not only makes information available, it also handles the retrieval of the resource. RLUS (Retrieval) would probably be a direct information broker.</td>
</tr>
<tr>
<td>Healthcare Practitioner</td>
<td>Term used in the UK with similar scope to the broader meaning of Caregiver as above.</td>
</tr>
<tr>
<td>Indirect Information Broker</td>
<td>See Information Broker. An indirect information broker makes information available about a resource, but leaves the retrieval of that object to another system. RLUS (Location) might be part of an indirect information broker.</td>
</tr>
<tr>
<td>Information</td>
<td>A system that mediates information exchanges. There are generally two aspects of brokering: location and fulfillment. Location would show that a resource is available. Fulfillment would</td>
</tr>
<tr>
<td><strong>Broker</strong></td>
<td>retrieve a chosen resource, for example, or manage a transaction involving that resource. A good example is a stock broker, who firsts identifies securities that may be exchanged, and then fulfills a request by buying or selling these securities, completing any legal components involved with transfer of ownership, and notifying the customer that the transaction has been completed.</td>
</tr>
<tr>
<td><strong>Interface specification</strong></td>
<td>A statement of what an architect chooses to make known about an element in order for other entities to interact or communicate with it (source: SEI Software Architecture Glossary)</td>
</tr>
<tr>
<td><strong>Introspective Interfaces</strong></td>
<td>By calling the introspective interface, clients can access information that helps them to control how they use the component and/or monitor and reason about it. General metadata provided by an introspective interface could include the component's type, name, location, version, interfaces, configuration and deployment parameters, as well as its dependencies to other components. An introspective interface could also provide metadata useful for managing the component, such as the number of clients, load, and resource usage. An introspective interface supports calling a component through a Dynamic Invocation Interface. An Introspective Interface provides controlled access to a component's details without breaking its encapsulation. In addition, it separates component usage from the process of obtaining information about it, which makes it ideal for tools to obtain information without becoming dependent on their specific interfaces, internal design, and implementation details. (source: Frank Buschmann - Kevlin Henney - Douglas C. Schmidt, <em>Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing</em>, 4th Volume, Wiley, 2007)</td>
</tr>
<tr>
<td><strong>Localized Information Model (LIM)</strong></td>
<td>LIMs allow additional constraints of information models without impacting the interoperability of instances with other instances conformant to the (more generic) parent Constrained Information Model (CIM). LIMs allow for semantic signifiers to be shared between interoperable instances of services, and are mandated as the mechanism for sharing semantic signifiers when one or more of the service instances is RLUS compliant. LIMs are, by definition, serializable and expressable through UML.</td>
</tr>
<tr>
<td><strong>Provider</strong></td>
<td>A provider of a service delivers the results of the service interaction. A provider will normally be the one that responds to the service interaction. Provider interfaces are used as the type of a Service Contract and are bound by the terms and conditions of that service contract. (source: OMG, <em>Service oriented architecture Modeling Language (SooML) - Specification for the UML Profile and Metamodel for Services (UPMS), 1.0</em>)</td>
</tr>
<tr>
<td><strong>Query by Example</strong></td>
<td>In a Query by Example scenario, a valid resource is expected by the RLUS implementation as query input. The RLUS instance validates the document against the appropriate Semantic Signifier. In executing the search, the underlying RLUS service implementation is expected to find corresponding Resource/s whose elements match the provided input document with an evaluation centered on the “AND” logical operand.</td>
</tr>
<tr>
<td><strong>Registry</strong></td>
<td>A registry is understood to be a source of meta-information regarding the identification and location of actual data. A Registry does not contain the data but only the information about the data (metadata). An example of a Standard registry is UDDI (that exposes information about services) while the ebXML RegRep Standard (that is designed to expose information about complex business services and agreements) has the capability of a registry and of a repository.</td>
</tr>
<tr>
<td><strong>Repository</strong></td>
<td>A repository is understood to be a place where electronic data is stored. It could be a relational structure, an object-oriented structure, or a file system, depending on the deployment context.</td>
</tr>
<tr>
<td>Resource</td>
<td>Any asset, information, data, or object that is able to be modeled in a Semantic Signifier and which serves a business requirement by being exposed through an RLUS-compliant interface. More concretely, a resource can be any intelligible and identifiable entity in a context, as a business object, a document and in general any identifiable set of information useful in a specific business context. The different conformance profiles allow an organization to choose whether the object itself or a representation is made available through the service. Different semantic signifiers allow similar functionality to be extended to different models of information in a consistent and comprehensible manner. Semantic profiles aggregate semantic signifiers in order to describe the information available through the service. Therefore, the scope of information can be described explicitly.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>RLUS</td>
<td>Retrieve, Locate, and Update Service</td>
</tr>
<tr>
<td>RLUS System</td>
<td>Any implementation that uses a Normative RLUS interface specification defined by RLUS (SFM and STMs)</td>
</tr>
<tr>
<td>Semantic Signifier</td>
<td>Semantic Signifiers are identifiers of information constructs that specify the structure and meaning of data. See Section § 5.3 Semantic Profiles and Semantic Signifier for an extended description of Semantic Signifiers.</td>
</tr>
<tr>
<td>Service Contract</td>
<td>A Service Contract is the specification of the agreement, between providers and consumers, of a service including what information, products, assets, value and obligations will flow between the providers and consumers of that service – it specifies the service without regard for realization, capabilities or implementation. A Service Contract does not require the specification of who, how or why any party will fulfill their obligations under that Service Contract, thus providing for the loose-coupling of the SOA paradigm. In most cases a Service Contract will specify two roles (provider and consumer) – but other service roles may be specified as well. (From OMG, Service oriented architecture Modeling Language (SoaML) - Specification for the UML Profile and Metamodel for Services (UPMS), 1.0)</td>
</tr>
<tr>
<td>Service Level Agreement (SLA)</td>
<td>An agreement between a service consumer and a service provider regarding which services will be delivered and the measurable levels of those services the provider is expected to achieve.</td>
</tr>
</tbody>
</table>
Appendix B: Profiles Examples

This Appendix provides some examples of possible Conformance Profiles that can be specified in a RLUS implementation. The profiles discussed here are only basic examples. The RLUS interface can support different content types, including but not limited to documents, and different vocabularies such as HL7 v3 and HL7 v2.x.

Appendix B.1: CDAr2 profiles

This example uses CDAr2 as the Semantic Signifier. The specific Semantic Signifier can be based on the normative XSD or on a constrained template (as a HL7-CCD, IHE-XDS-LAB or a local template).

The functional profile can use the appropriate set of standard operations. In this profile the header of a CDAr2 resource is used as an implicit metadata set (specified as a searchable fragment).

As an example this profile is a simple way to interface systems that expose and receive a CDAr2 document of various types. Behind the interface, as an example, a set of XDS implementations can exist but the communication can be simplified for a consumer. It’s not required to burden an RLUS consumer with knowledge of the ebXML registry but only CDA and common WS* technology.

These kind of issues are recognized by the IHE whitepaper on SOA and therefore it realizes a clean architecture without a complex mix of different XML vocabulary (HL7 - v2 and v3 - and ebXML) and dependence on a specific implementation (ebXML Registry Repository v3.0).

---


33 The HL7 CDA 2 R-MIM diagram is inserted here as a visual placeholder for the R-MIM included in the CDA Release 2 standard (CDA R-MIM POCD_RM000040).

34 IHE, A Service-Oriented Architecture (SOA) View of IHE Profiles, IT Infrastructure White Paper, September 28, 2009
This service architecture is presented in Figure 23.

![RLUS Gateway Services Architecture with a CDAr2 profile](image)

It's relevant to stress the point that with RLUS the internal implementation architecture is hidden so it's possible to mix different implementations in a RLUS network. Different deployment scenario can be used and mixed without impact (see § 2.1.3).

---

As in XDS and XCA and in general in XD* family
Appendix B.2: RLUS Registry Repository Profile with RCMR and CDAr2

This profile example uses HL7 version 3 Medical Record Domain (RCMR) to implement a RLUS Registry/Repository architecture. The Medical Records domain currently supports clinical document management and has a structure substantially based on the CDAr2 header. Below in Figure 24, the DMIM of RCMR that should be the basis for this profile specification is described. This is an example of a semantic profile not directly based on documents: the resource is mainly the metadata not a document in itself.

Figure 24: RCMR D-MIM

A variation of use of an RCMR Semantic Signifier is a Registry and Repository architecture (similar to IHE-XDS). In this scenario an RCMR Semantic Signifier is used in conjunction with a CDAr2 Semantic Signifier (or other types of Semantic Signifiers) to realize an architecture where a central Registry exposes the metadata and the resource locations and the resources are exposed by simplified RLUS instances. In this scenario the

The HL7 v3 version considered here is the Normative 2011.

The HL7 RCMR D-MIM diagram is inserted here as a visual placeholder for the D-MIM included in the HL7 V3 standard (Medical Records RCMR_DM000050UV).
Figure 25 describes this possible service architecture that supports a deployment scenario as described in § 2.1.3.

---

38 See Glossary (Appendix A) for definition of *Indirect Information Broker* term.

39 See Glossary (Appendix A) for definition of *Direct Information Broker* term.
Appendix C: HL7 EHR Functional Model Traceability

The functionality defined in the EHR-S model doesn’t have a one-to-one exact mapping into the functionality defined by the RLUS service. However the functionality provided by RLUS can be seen as basic infrastructure that supports the implementation of EHR-S functionality.

For example, function DC.1.1.3.2 Manage medication list needs to access recently updated medication lists in other organizations as well as pharmacy dispense records in order to form a sufficient base for the current medication list. The RLUS doesn’t fully implement the function DC.1.1.3.2, but it can be used to implement parts of that function.

<table>
<thead>
<tr>
<th>EHR Function ID</th>
<th>EHR Function Name</th>
<th>EHR Function Statement</th>
<th>Supporting RLUS capabilities</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC.1.1.3</td>
<td>Manage summary lists</td>
<td>Create and maintain patient-specific summary lists that are structured and coded where appropriate.</td>
<td>Runtime Capabilities Administrative Capabilities Introspective Capabilities</td>
<td>Patient summary lists can be created from patient specific data and displayed and maintained in a summary format. The functions below are important, but do not exhaust the possibilities.</td>
</tr>
<tr>
<td>DC.1.1.3.1</td>
<td>Manage problem list</td>
<td>Create and maintain patient-specific problem lists.</td>
<td>Runtime Capabilities Administrative Capabilities Introspective Capabilities</td>
<td>A problem list may include, but is not limited to: Chronic conditions, diagnoses, or symptoms, functional limitations, visit or stay-specific conditions, diagnoses, or symptoms. Problem lists are managed over time, whether over the course of a visit or stay or the life of a patient, allowing documentation of historical information and tracking the changing character of problem(s) and their priority. All pertinent dates, include date noted or diagnosed, dates of any changes in problem specification or prioritization, and date of resolution are stored. This might include time stamps, where useful and appropriate. The entire problem history for any problem in the list is viewable.</td>
</tr>
<tr>
<td>DC.1.1.3.2</td>
<td>Manage medication list</td>
<td>Create and maintain patient-specific medication lists.</td>
<td>Runtime Capabilities Administrative</td>
<td>Medication lists are managed over time, whether over the course of a visit or stay, or the lifetime of a patient. All pertinent dates, including medication start,</td>
</tr>
</tbody>
</table>
### DC.1.1.3.3 Manage allergy and adverse reaction list

- **Capabilities**
  - Introspective Capabilities
- **Runtime Capabilities**
  - Administrative Capabilities
  - Introspective Capabilities

**Description:** Allergens, including immunizations, and substances are identified and coded (whenever possible) and the list is managed over time. All pertinent dates, including patient-reported events, are stored and the description of the patient allergy and adverse reaction is modifiable over time. The entire allergy history, including reaction, for any allergen is viewable. The list(s) include drug reactions that are not classifiable as a true allergy and intolerances to dietary or environmental triggers. Notations indicating whether item is patient reported and/or caregiver verified are supported.

### DC.1.1.5 Summarize health record

- **Capabilities**
  - Introspective Capabilities
- **Runtime Capabilities**
  - Administrative Capabilities

**Description:** A key feature of an electronic health record is its ability to present, summarize, filter, and facilitate searching through the large amounts of data collected during the provision of patient care. Much of this data is date or date-range specific and should be presented chronologically. Local confidentiality rules that prohibit certain users from accessing certain patient information must be supported.

### DC.1.1.6 Manage clinical documents and notes

- **Capabilities**
  - Introspective Capabilities
- **Runtime Capabilities**
  - Administrative Capabilities

**Description:** Clinical documents and notes may be created in a narrative form, which may be based on a template. The documents may also be structured documents that result in the capture of coded data. Each of these forms of clinical documentation is important and appropriate for different users and situations.
| DC.1.1.7 | Capture external clinical documents | Incorporate clinical documentation from external sources. | Runtime Capabilities Advanced Capabilities Introspective Capabilities | Mechanisms for incorporating external clinical documentation (including identification of source) such as image documents and other clinically relevant data are available. Data incorporated through these mechanisms is presented alongside locally captured documentation and notes wherever appropriate.

| DC.1.1.8 | Capture patient-originated data | Capture and explicitly label patient-provided and patient-entered clinical data, and support provider authentication for inclusion in patient history | Runtime Capabilities Administrative Capabilities Introspective Capabilities | It is critically important to be able to distinguish patient-provided and patient-entered data from clinically authenticated data. Patients may provide data for entry into the health record or be given a mechanism for entering this data directly. Patient-entered data intended for use by caregivers will be available for their use.

| DC.1.5.1 | Manage consents and authorizations | Create, maintain, and verify patient treatment decisions in the form of consents and authorizations when required. | Runtime Capabilities | Treatment decisions are documented and include the extent of information, verification levels and exposition of treatment options. This documentation helps ensure that decisions made at the discretion of the patient, family, or other responsible party govern the actual care that is delivered or withheld.

| DC.1.5.2 | Manage patient advanced directives | Capture, maintain and provide access to patient advance directives. | Runtime Capabilities | Patient advance directives and caregiver DNR orders can be captured as well as the date and circumstances under which the directives were received, and the location of any paper records of advance directives as appropriate.

| DC.3.2.2 | Pharmacy communication | Provide features to enable secure bidirectional communication of information electronically between practitioners and pharmacies or between practitioner and intended recipient of pharmacy orders. | Runtime Capabilities Administrative Capabilities | When a medication is prescribed, routed to the pharmacy or another intended recipient of pharmacy orders. This information is used to avoid transcription errors and facilitate detection of potential adverse reactions. Upon filling the prescription, information is sent back to the practitioner to indicate that the patient received the medication. If there is a question from the pharmacy, that communication can be presented to the caregiver with their other tasks.
<table>
<thead>
<tr>
<th>DC.3.2.5</th>
<th>Communication with medical devices</th>
<th>Support communication and presentation of data captured from medical devices.</th>
<th>Runtime Capabilities</th>
<th>Communication with medical devices is supported as appropriate to the care setting. Examples include: vital signs/pulse-oximeter, anesthesia machines, home diagnostic devices for chronic disease management, laboratory machines, and bar coded artifacts (medicine, immunizations, demographics, history, and identification).</th>
</tr>
</thead>
<tbody>
<tr>
<td>S.2.2</td>
<td>Report generation</td>
<td>Provide report generation features for the generation of standard and ad hoc reports.</td>
<td>Runtime Capabilities</td>
<td>A user can create standard and ad hoc reports for clinical, administrative, and financial decision-making, and for patient use - including structured data and/or unstructured text from the patient’s health record. Reports may be linked with financial and other external data sources (i.e. data external to the entity). Such reports may include patient-level reports, caregiver/facility/delivery system-level reports, population-level reports, and reports to public health agencies. Examples of patient-level reports include: administratively required patient assessment forms, admission/transfer/discharge reports, operative and procedure reports, consultation reports, and drug profiles. Examples of population-level reports include: reports on the effectiveness of clinical pathways and other evidence-based practices, tracking completeness of clinical documentation, etcetera. Examples of reports to public health agencies include: vital statistics, reportable diseases, discharge summaries, immunization data including adverse outcomes, cancer data, and other such data necessary to maintain the publics’ health (including suspicion of newly emerging infectious disease and non-natural events).</td>
</tr>
<tr>
<td></td>
<td>Health record output</td>
<td>Allowing users to define the records and/or reports that are considered the formal health record for disclosure purposes, and provide a mechanism for both chronological and specified record element output.</td>
<td>Runtime Capabilities</td>
<td>Provide hardcopy and electronic output that can fully chronicles the healthcare process, supports selection of specific sections of the health record, and allows healthcare organizations to define the report and/or documents that will comprise the formal health record for disclosure purposes.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>S.3.2</td>
<td>Information access for supplemental use</td>
<td>Support extraction, transformation and linkage of information from structured data and unstructured text in the patient’s health record for care management, financial, administrative, and public health purposes.</td>
<td>Runtime Capabilities</td>
<td>Using data standards and technologies that support interoperability, information access functionalities serve primary and secondary record use and reporting with continuous record availability and access that ensure the integrity of (1) the health record, (2) public health, financial and administrative reporting, and (3) the healthcare delivery process</td>
</tr>
<tr>
<td>S.3.3.4</td>
<td>Support of service requests and claims</td>
<td>Support interactions with other systems, applications, and modules to support the creation of health care attachments for submitting additional clinical information in support of service requests and claims.</td>
<td>Runtime Capabilities</td>
<td>Automatically retrieves structured data, including lab, imaging and device monitoring data, and unstructured text based on rules or requests for additional clinical information in support of service requests or claims at the appropriate juncture in the encounter workflow</td>
</tr>
<tr>
<td>I.2</td>
<td>Health record information and management</td>
<td>Manage EHR information across EHR-S applications by ensuring that clinical information entered by providers is a valid representation of clinical notes; and is accurate and complete according to clinical rules and tracking amendments to clinical document. Ensure that information entered by or on behalf of the patient is accurately represented.</td>
<td>Runtime Capabilities</td>
<td>Since EHR information will typically be available on a variety of EHR-S applications, an EHR-S must provide the ability to access, manage and verify accuracy and completeness of EHR information, and provide the ability to audit the use of and access to EHR information.</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Runtime Capabilities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>---------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>I.2.1</strong> Data Retention and Availability</td>
<td>Retain, ensure availability, and destroy health record information according to organizational standards. This includes: &gt; Retaining all EHR-S data and clinical documents for the time period designated by policy or legal requirement; &gt; Retaining inbound documents as originally received (unaltered); &gt; Ensuring availability of information for the legally prescribed period of time; and &gt; Providing the ability to destroy EHR data/records in a systematic way according to policy and after the legally prescribed retention period.</td>
<td>Discrete and structured EHR-S data, records and reports must be: &gt; Made available to users in a timely fashion; &gt; Stored and retrieved in a semantically intelligent and useful manner (for example, chronologically, retrospectively per a given disease or event, or in accordance with business requirements, local policies, or legal requirements); &gt; Retained for a legally-prescribed period of time; and &gt; Destroyed in a systematic manner in relation to the applicable retention period. An EHR-S must also allow an organization to identify data/records to be destroyed, and to review and approve destruction before it occurs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>I.2.4</strong> Extraction of health record information</td>
<td>Manage data extraction in accordance with analysis and reporting requirements. The extracted data may require use of more than one application and it may be pre-processed (for example, by being de-identified) before transmission. Data extractions may be used to exchange data and provide reports for primary and ancillary purposes.</td>
<td>An EHR-S enables an authorized user, such as a clinician, to access and aggregate the distributed information, which corresponds to the health record or records that are needed for viewing, reporting, disclosure, etc. An EHR-S must support data extraction operations across the complete data set that constitutes the health record of an individual and provide an output that fully chronicles the healthcare process. Data extractions are used as input to continuity of care records. In addition, data extractions can be used for administrative, financial, research, quality analysis, and public health purposes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>I.3.1</strong> Distributed registry access</td>
<td>Enable system communication with registry services through standardized interfaces and extend to services provided externally to an EHR-S.</td>
<td>An EHR-S relies on a set of infrastructure services, directories, and registries, which may be organized hierarchically or federated, that support communication between EHR-S’. For example, a patient treated by a primary care physician for a chronic condition may become ill while out</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
of town. The new caregiver’s EHR-S interrogates a local, regional, or national registry to find the patient’s previous records. From the primary care record, a remote EHR-S retrieves relevant information in conformance with applicable patient privacy and confidentiality rules. An example of local registry usage is an EHR-S application sending a query message to the Hospital Information System to retrieve a patient’s demographic data.
Appendix D: Mapping with Service Technical Model (STM)

In this Appendix we present the mapping of RLUS SFM with the current version of RLUS Platform Independent Model presented in the OMG Retrieve, Locate and Update Service Specification.  

The current version includes a WS* Platform Specific Model (PSM) and a RESTful PSM is emerging (RLUS hDATA Restful Transport).

In Figure 26, a SoaML Service Architecture Diagram of the current RLUS PIM.

Figure 26: RLUS STM PIM Service Architecture

Actually the RLUS PIM (in v 1.0.1) is composed of two interfaces:

- RLUS Management and Query Interface
- RLUS Semantic Profiles Interface

Figure 27, a SoaML Service Structure Diagram, details the service contracts defined by the RLUS STM Interfaces with the operations specified.

Consider that actually only the Management and Query Interface is specified in a WS* PSM. The Semantic Profiles Interface is specified only at the PIM level as an explanation of optional features of an RLUS implementation.

---

40 The current version is the 1.0.1 (formal/2011-07-02). See: http://www.omg.org/spec/RLUS/
Figure 27: RLUS STM PIM Services Contracts

Figure 28 represents the mapping between the RLUS capabilities, described here in the RLUS SFM, and the interface specified by The RLUS STM PIM (v 1.0.1).
The RLUS PIM, for pragmatic reasons, includes the describe() operation in the RLUS Management and Query Interfaces.
### Appendix E: Relevant Standards and Profiles

(Informative)

<table>
<thead>
<tr>
<th>Related Standard and Profiles</th>
<th>Relationship</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ebXML Registry Repository v3.0.1</td>
<td>Relevant External Work – The ebXML Registry Repository has been designed as a part of the ebXML standard to support business discovery and agreement about ebXML services. In some aspects it’s similar to UDDI but more sophisticated and complex. The ebXML Registry Repository, for the registry features, is profiled by IHE-XDS. In the RLUS context, an ebXML registry, theoretically, can be deployed behind an RLUS interface.</td>
<td><a href="http://www.oasis-open.org">www.oasis-open.org</a></td>
</tr>
<tr>
<td>HL7 CDA 2</td>
<td>The HL7 CDA Specification may be used as a structure for the payload definition of RLUS-retrieved results. In other words, the parameters on the RLUS service interface may use CDA-conformant representations as the structure and semantic of the data it is managing.</td>
<td><a href="http://www.hl7.org">www.hl7.org</a></td>
</tr>
<tr>
<td>HL7 EHR-S</td>
<td>See: <a href="http://www.hl7.org/ehr">www.hl7.org/ehr</a></td>
<td><a href="http://www.hl7.org/ehr">www.hl7.org/ehr</a></td>
</tr>
<tr>
<td>HL7 Version 3 Reference Information Model (HL7 V3 RIM)</td>
<td>The HL7 Reference Information Model provides the underpinning for the information semantics that are used in HL7-conformance profiles of the RLUS specification. For these profiles, the RIM and other RIM-derived information models identify the data elements, data types, structure, and underlying terminologies for payload crossing the RLUS service interfaces.</td>
<td><a href="http://www.hl7.org">www.hl7.org</a></td>
</tr>
<tr>
<td>IHE XDS Cross Enterprise Document Exchange</td>
<td>Relevant External Work – IHE-XDS is an IHE profile based on the ebXML Registry Repository standard. IHE XDS can be considered as an example of an early RLUS implementation that partially conforms to an RLUS HL7 CDA Profile but it is dependent on an ebXML Registry Repository-specific implementation. In practice, both the RLUS standard and XDS partially share similar capabilities. As both standards evolve, their harmonization will become an ongoing effort for both communities.</td>
<td><a href="http://www.ihe.net">www.ihe.net</a></td>
</tr>
<tr>
<td>IHE XCA Cross-Community Access</td>
<td>Relevant External Work – IHE-XCA is an IHE profile based on the ebXML Registry Repository standard used as a cross community XDS gateway. The IHE XCA profile partially conforms to an RLUS HL7 CDA Profile but it’s dependent on an ebXML Registry Repository-</td>
<td><a href="http://www.ihe.net">www.ihe.net</a></td>
</tr>
</tbody>
</table>
specific implementation and the exposed interfaces do not realize true information hiding. So the interface is slightly different from XDS and the internal XDS architecture is exposed.

As both standards evolve, their harmonization will become an ongoing effort for both communities.

### Localized Information Model (LIM)

The HL7 Organization's Template Special Interest Group has undertaken to provide for the description and localization of information models. LIMs provide a way to communicate the informational semantics of an RLUS instance to trading partners. See HL7 Templates below.

### Universal Description, Discovery, and Integration

**Relevant External Work** - UDDI provides a platform for Discovery and Description of Services, and helped to broadly define the business needs for true automated service-to-service RLUS interactions. The UDDI specification informs the RLUS SFM in its notion of topologies and in its design for automated discovery and description. Thus, it defined appropriate functional boundaries and expectations without creating a normative concept.

### HL7 Templates

The HL7 Templates Special Interest Group (Templates SIG) is currently in the process of harmonizing requirements from among the CEN, OpenEHR, and HL7 communities. Each of these communities is using some form of structure, constraint, and semantics to do precisely the types of representations and uses that are expected of RLUS semantic signifiers.

---

www.hl7.org

www.oasisopen.org

www.hl7.org