HP Systinet

Software Version: 10.01
Windows and Linux Operating Systems

Concepts Guide

Document Release Date: June 2015
Software Release Date: June 2015

Legal Notices

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: **http://h20230.www2.hp.com/selfsolve/manuals**

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to: **http://h20229.www2.hp.com/passport-registration.html**

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Online web site at: **http://www.hp.com/go/hpsoftwaresupport**

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

HP Software Solutions Now accesses the HPSW Solution and Integration Portal Web site. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this Web site is **http://h20230.www2.hp.com/sc/solutions/index.jsp**

About this PDF Version of Online Help

This document is a PDF version of the online help. This PDF file is provided so you can easily print multiple topics from the help information or read the online help in PDF format. Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not be present in this PDF version. Those topics can be successfully printed from within the online help.

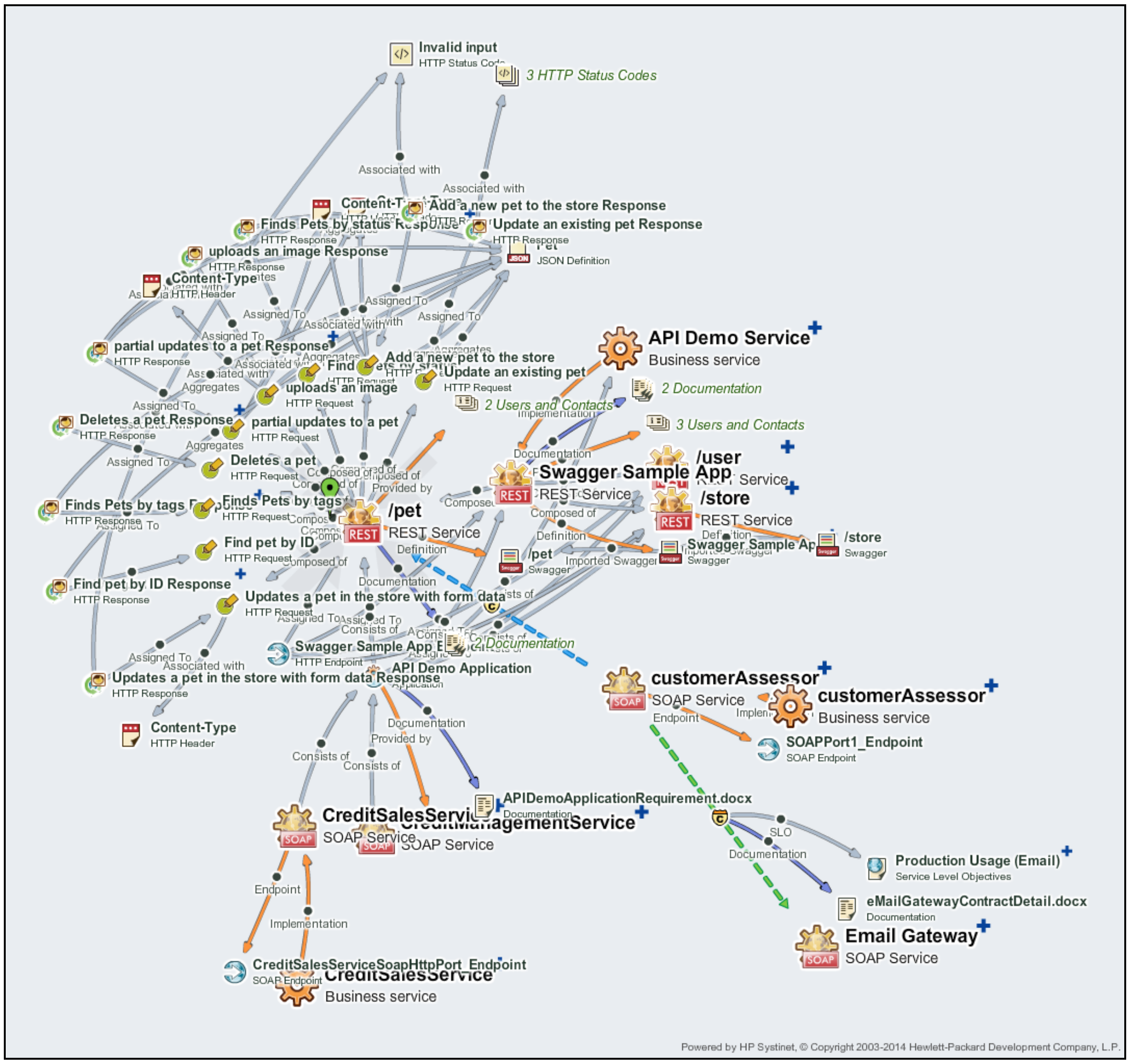
Contents

Chapter 1: Concept overview

This guide collects all the product concepts together into a single PDF document.

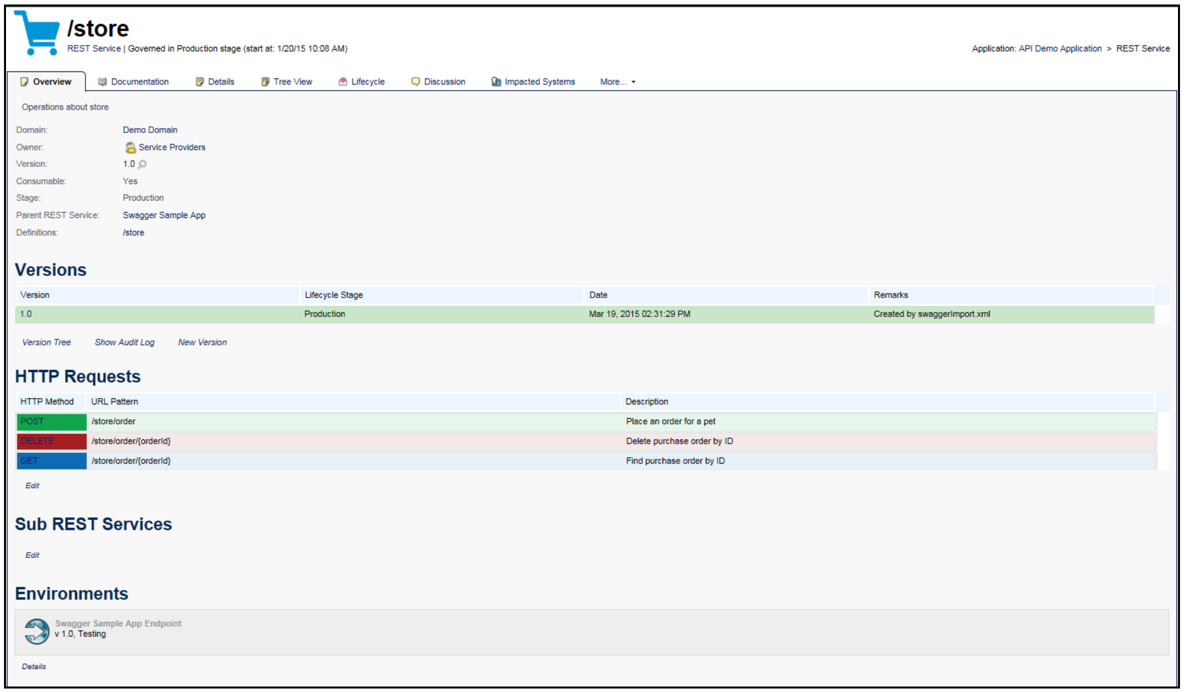
Chapter 2: API Management

API (Application Programming Interface) Management is a new feature that makes Systinet 10.01 capable of authoring and collaborating REST services in a similar manner to those of web services.



REST service is a new feature for API Management and can be created by the following ways:

- **Manual entry**: Systinet 10.01 User Interface (UI) contains new components designed specifically for REST services. UI of REST services is customizable and could be reused for other artifact types.



/store

REST Service | Governed in Production stage (start at: 1/20/15 10:08 AM)

Application: API Demo Application > REST Service

Overview | Documentation | Details | Tree View | Lifecycle | Discussion | Impacted Systems | More...

Operations about store

Domain:	Demo Domain
Owner:	Service Providers
Version:	1.0
Consumable:	Yes
Stage:	Production
Parent REST Service:	Swagger Sample App
Definitions:	/store

Versions

Version	Lifecycle Stage	Date	Remarks
1.0	Production	Mar 19, 2015 02:31:29 PM	Created by swaggerImport.xml

Version Tree   Show Audit Log   New Version

HTTP Requests

HTTP Method	URL Pattern	Description
POST	/store/order	Place an order for a pet
DELETE	/store/order/{orderId}	Delete purchase order by ID
GET	/store/order/{orderId}	Find purchase order by ID

Edit

Sub REST Services

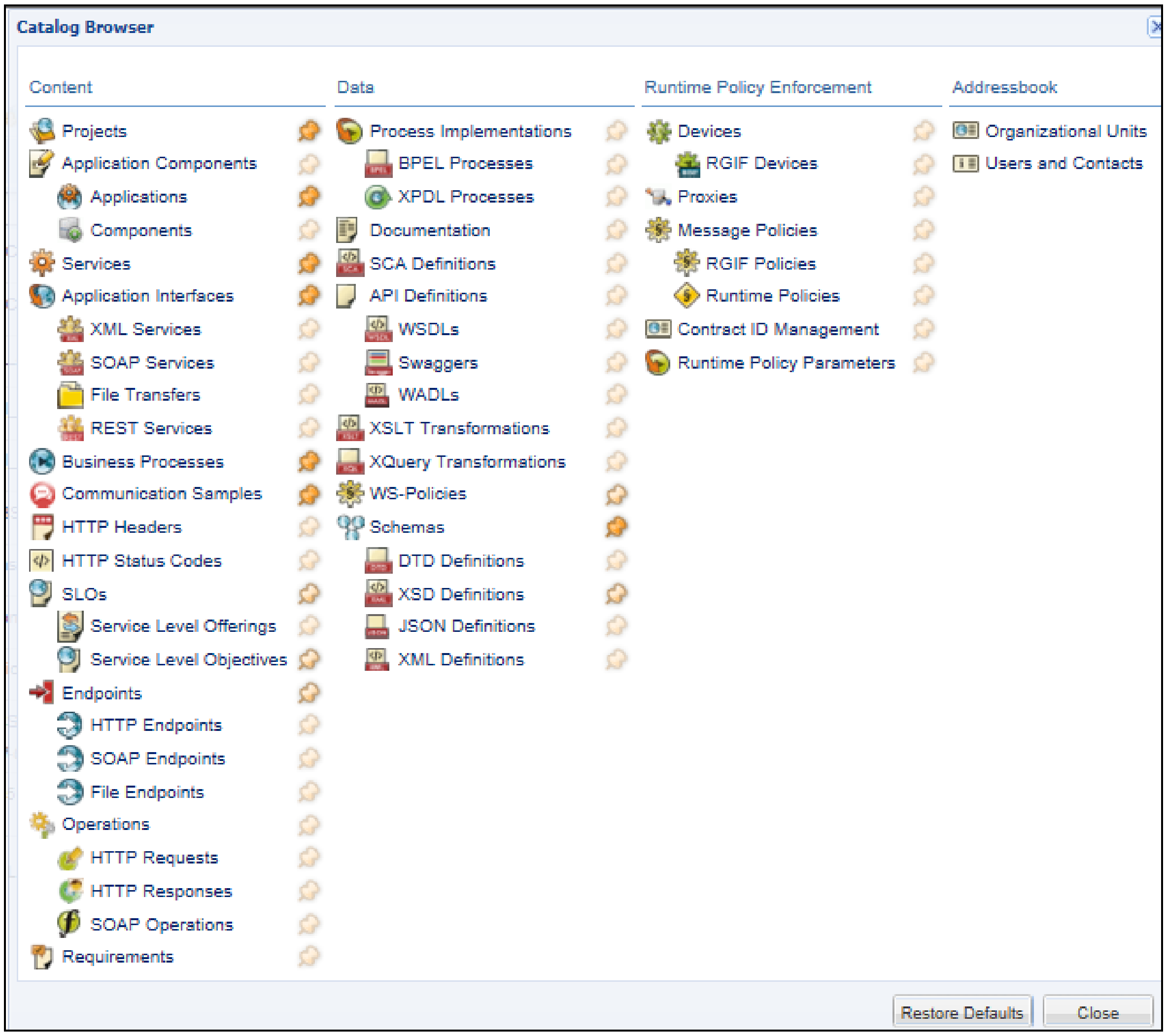
Edit

Environments

Swagger Sample App Endpoint
v 1.0, Testing

Details

- **CSV import**: Systinet 10.01 contains bug fixes for CSV import.
- **Swagger import**: Swagger format importing is supported for out-of-the-box (OOTB) and is customizable.
- **WADL import**: WADL format importing is supported for OOTB and is customizable.
- **Discovered artificats**: REST services could be discovered from runtime Policy Enforcement Points (PEP) as an effort to reverse engineer services model from Run-time to Design-time.
- **New artifacts types**: New artifact types are introduced for API Management.



SOA Governance

SOA is no longer an emerging technology but a standard that is a routine part of application development. Currently, most large organizations using SOA in some form are having issues dealing with or transitioning to SOA. Occasionally, these organizations resolve the problems with SOA without explicitly knowing how or why. They often struggle to organize their IT applications, services, or software development departments and system integration. It is clear that in large multinational companies that this can be a difficult task requiring a comprehensive software solution.

SOA is an architectural concept wherein the basic building blocks of software development are services. A system or application consists of a set of autonomous services communicating with each other. Service-oriented architecture is about designing and developing business functionality built on the basis of shared reusable services. By combining these services, an organization can effectively respond to new demands and deliver new functionality better, faster, and cheaper. An important concept here is reusability. Individual modules or components within large-scale information systems may have the same functionality for different implementations and are integrated into systems designed to obtain the finished, certified, and trusted software service.

Over time, the number of services grows and the relationships and dependencies between them become increasingly complex. In the event of a failure of just one service, the entire functionality of the system may become compromised. Not only must an organization have an overview of the developed, offered, and deployed applications and services but must be able to effectively manage this growing infrastructure. Is there a service you need and how to you find it? Who maintains the service and what is its status? What happens when the service is altered? How can I ensure that the services used have the necessary certification? How to ensure that services are in line with both IT and business policies? These are precisely the issues addressed by SOA governance.

SOA governance manages best practices, service development, relationships, lifecycle management, specific quality requirements, the definition and enforcement of policies (polices - rules, restrictions, regulations) that services must meet during lifecycle stages either during the design stage or while running their own services. An important aspect is that of contract management, where it is possible to define the conditions and relationships between the providers and consumers of services. All these procedures may be operated manually or semi-automatically on the basis of written directives, instructions, sent spreadsheets, documents, or emails. However, with a good infrastructure, SOA lifecycle processes facilitate the deployment and operation of SOA within an organization which can be fully automated.

Systinet provides the foundation for SOA governance through its powerful SOA repository, and functionality and features focusing on the real problems in implementing SOA. Systinet is able to integrate with other products from the HP portfolio either in the field of quality management (Quality Center), or the running and monitoring the performance of services or applications from the perspective of end customers (HP Business Service Management).

SOA Repository

The basic software for automated SOA governance is usually an SOA repository. The repository serves as the primary resource for data source metadata (data about data) and other related information and not only for provided services, but also the complete SOA infrastructure. The structure of the content repository is typically defined with an extensible Service Definition Model (SDM). SDM defines a schema model, reflecting the SOA in the organization. The model includes business artifacts (entities), higher abstractions (Services, Business Processes, Applications, etc.), and technical artifacts (Implementations, Operations, Endpoints, XML Schema, WSDLs, etc.) It is important that the definition model is flexible and expandable so that it can effectively adjust to the environment of the organization. Information relevant to governance becomes part of the SOA data model definition. The metadata contains information needed for the creation, searching, validation, and approval of services in different stages of their lifecycle.

In principle, one can say that there are three types of metadata: business information, technical information, and information governance. Business information may be the type of service (application, infrastructure, business services), or the area on which services are focused (banking, health care, etc). Technical information would be service interfaces, security, protocols, etc. Governance information is primarily policy, lifecycle status, and relationships between services.

For details about the Systinet repository and SOA model, see "Catalog" on page 15 and "Service Definition Model (SDM)" on page 17.

Product Services

The main function of the Catalog is to manage content in the SOA repository. Users in specific roles can use and access the catalog in a different ways. Consumers of services are likely to be most interested in the services which provide required functionality under given conditions, and with required quality. In contrast, service providers want to view a service in the catalog or simply create a register with all pertinent details and offer it to potential consumers. Implementation of catalog content should be automated where possible, particularly with regard to the technical and implementation details of services. For example, an existing web service described by a WSDL is registered in the system by uploading a WSDL. The system then loads the necessary information and creates a breakdown of the service according to the SOA model.

For details about discovering, publishing, and maintaining content in Systinet, see "Search and Browse" on page 15, "Navigator" on page 20, "Authoring" on page 27, and "Artifact Management" on page 41.

Collaboration

In large organizations, it is important to provide a platform that enables users to quickly find the information relevant to them and to enable them to easily share their work with their colleagues. For example, users need to know about changes to services that affect them.

For details about collaboration, see "Collaboration" on page 26.

Lifecycle Management

Services and applications have several lifecycle stages. Each stage has its specific features and important aspects. Each organization may have a set of specific stages requiring different levels of detail. A typical service lifecycle may consist of an initial stage defining the service and giving the combined requirements, with subsequent phases are developing, testing, production, and eventually the service is retired or replaced by a new version or a completely new service. From a larger perspective, this can be divided into stages of development (design time) and production (runtime). To enable a service to move from one stage to another, you need to ensure that the service has the necessary requirements and quality standards completed. In the Catalog, a service is described by a set of artifacts organized into a lifecycle tree, which can be defined for each stage of the lifecycle process. The process of moving between stages is associated with a request for approval and automatic actions are defined, performed, and controlled to alter the artifacts based on the transition.

For more details about lifecycle governance and version management, see "Lifecycle" on page 61 and "Versioning" on page 47.

Lifecycle processes are governed by the set of business rules defined in SOA policies. Service conformity is automatically checked and the rules can be enforced on the services. If a service does not meet the required rules, a remedy may be required. From a technical point of view, policies can be defined using the standard WS-Policy (Web Service Policy). During the initial stages of a service lifecycle, there should be an analysis of requirements and use cases, and a contact or functional architect. In subsequent stages the technical analysis, the means of implementation, interface, operations, testing, access addresses, and of course, other contacts (administrator, architect, project manager, etc.) are important. At the time of contract creation, it is again important to verify whether the service is defined by an SLO (Service Level Objective). When services are in production, other interesting parameters, such as safety, methods of verifying identity and authorization policies for monitoring load, permeability, content, or context-dependent routing may become important.

For more details about policy, see "Policy Management" in the *Concepts Guide*.

Contract Management

Contract Manager enables the management of the relationships between providers and consumers of services. The process works so that consumers find services they would like to use and can make requests for their use with a defined level of service (SLO). These SLOs are essentially measurable performance indicators, such as throughput, availability, response time, the expected utilization of services, etc. The service provider is automatically notified of the request and can either approve or reject it based on the requirement. The service provider can easily view their consumers, which is important in assessing service popularity and reusability as well as determining who will be affected by a change or disruption to a service. Runtime enforcement may also be based on contract management using the RGIF feature.

For more details about contracts, see "Contracts" on page 50 and "RGIF Overview" on page 1.

Integration

The possibility of simple integration with other systems is an important characteristic of every modern enterprise software company. In terms of the service lifecycle it is useful in the development stage to integrate development and testing tools. During service runtime organizations require access to information systems monitoring services in production environments. The repository can integrate with other systems for automated data publishing, retrieval, and repeated use. Process development services may also be connected to software project management tools. Integration may be broad, but it is necessary to have the appropriate tools and technologies enabling an effectively implemented integration. One example is the remote interface for access to the SOA repository interface exposed via REST API (Representational State Transfer Application Programming Interface). SOA artifacts in the REST architectural style represent source-specific data (resources) which can be accessed via HTTP basic operations for returning, creating, editing and deleting data.

For more details about integration, see "Product Integration" on page 71.

Reporting

Most organizations need frequent and accurate summaries of the status of their repository content. For example, the contractual status of services, or usage of services. Systinet provides an extensive set of default reports with additional customization options.

For more details about reporting, see "Reporting" on page 83.

Administration

The complexity of SOA governance can quickly become difficult to manage and can place large workloads on system administrators. Systinet provides comprehensive administration of users, their access to content and functionality, and the configuration of the product itself.

For more details about administration, see "Administration" on page 89.

Chapter 3: Search and Browse

Search and Browse takes advantage of the structure of the SDM Model to provide various search and browse methods, enabling you to locate the artifacts you want using the criteria that are most useful to you. For more details about the model, see "Service Definition Model (SDM)" on page 17.

The basic search input uses the common name and description properties of all artifacts to provide a method of searching the entire Catalog by the most widely known properties. You can expand this by adding additional criteria based on the artifact type, and specific properties of that artifact type. For more details, see "How to Search the Catalog" in the *User Guide*. For more details, see "Catalog" below.

Browsing allows you to start with a more specific criteria such as artifact type or categorization, list all the artifacts within that specific criteria and then narrow your search using the filtering functionality of tables. For more details, see "How to Browse by Artifact Type" on page 1 and "How to Browse by Categories" on page 1"How to Browse by Artifact Type" and "How to Browse by Categories" in the *User Guide*.

After locating the content you require, you can also access previously located content using Recent Documents, Favorites, and Stored Search functionality. For details, see "How to Use Favorites", "How to Access Recent Documents" and "How to Use Saved Searches" in the *User Guide*.

In addition to these search and browse methods, Systinet provides a visual representation of your Catalog content with the Graphical Navigator. For details, see "Navigator" on page 20.

HP Software also provides a set of IDE plugins with integrated search functionality. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

Systinet also provides collaborative features that are useful in search and browse use cases. For details, see "Collaboration" on page 26.

Catalog

The Catalog provides a single, central storage point enabling you to organize your data and metadata.

The Catalog stores and organizes your business data. The most important entities in your business, such as services and their implementations, users and groups, projects and business processes, are represented in the Catalog by artifact instances. The Catalog provide a central reference point for governance and provision. For more details about artifacts, see "Artifacts and Properties" on page 18.

The Catalog provides the following key features:

- **Visibility**

Systinet provides numerous features to explore the content of the Catalog, to visualize the relationships between artifacts, and to keep your colleagues up-to-date with the current status of your work.

- Locate Catalog content and its data and metadata using multiple search and browse methods. For details, see "Search and Browse" on the previous page.
- Visualize your Catalog content using the Graphical Navigator. For details, see "Navigator" on page 20.
- Collaborate with your colleagues. For details, see "Collaboration" on page 26.

- **Governance**

  Systinet is primarily a governance platform and provides extensive support for publishing, managing and distributing your content.

  - Author and publish your data content in the Catalog. For details, see "Authoring" on page 27.
  - Edit, categorize, and manage your content. For details, see "Artifact Management" on page 41.
  - Maintain multiple versions of your content. For details, see "Versioning" on page 47.
  - Control the consumption and re-use of your content using Contract Management. For details, see "Contracts" on page 50.
  - Ensure that content conforms to your business policies. For details, see "Policy Management" in the *Concepts Guide*.
  - Control the development of your catalog content using Lifecycle Governance. For details, see "Lifecycle" on page 61.
  - Utilize other products to propagate and enhance your content. For details, see "Product Integration" on page 71.

The content of the Catalog conforms to the Service Definition Model. The SDM Model defines an extensible hierarchy of artifact types, with defined properties and relationships. For more details, see "Service Definition Model (SDM)" on the next page.

Users access the Catalog through the following features of Systinet:

- Access and work with Catalog content using the Catalog Tab. For details, see "Catalog Tab" in the *User Guide*.
- Visualize Catalog content using the Navigator Tab. For details, see "Navigator Tab" in the *User Guide*.
- Administrators access and work with system data using the Administration Tab. For details, see the *Administration Guide*.
- Developers can access and work with Catalog content remotely using WebDav clients or Systinet

IDE Integrations. For details, see "WebDAV Compliant Publishing in the *Developer Guide* and *Plugin for Eclipse* and *Plugin for Visual Studio*.

Service Definition Model (SDM)

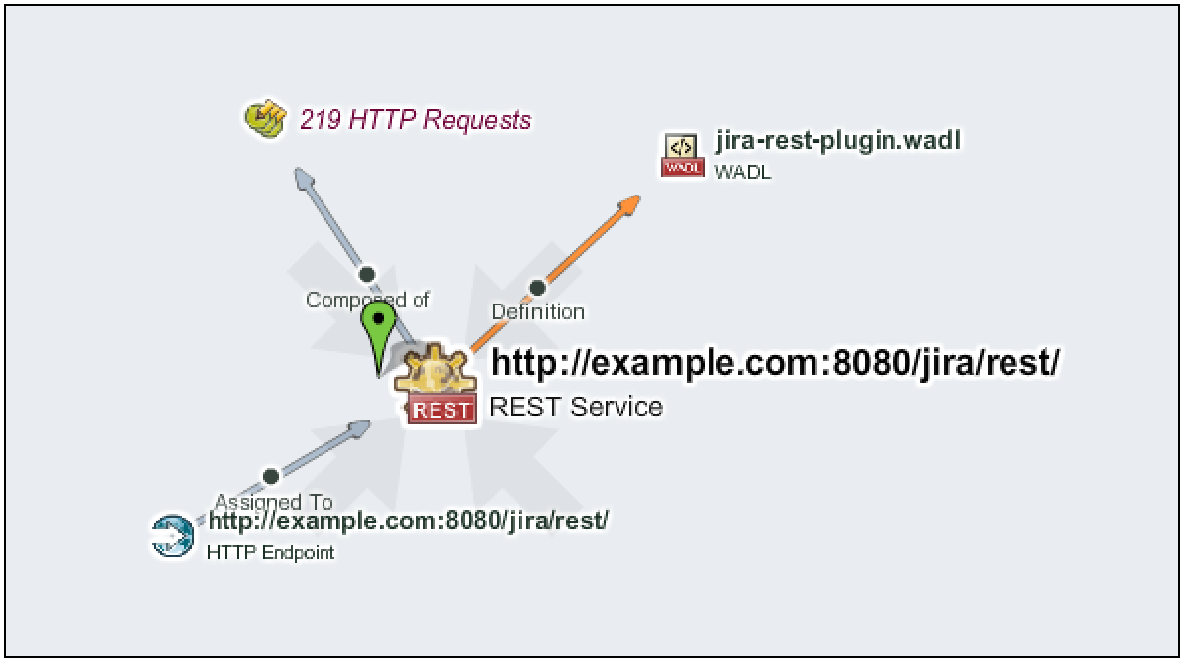
At the core of Systinet is the Service Definition Model (SDM). The SDM defines the Catalog data schema, reflecting the actual semantics of the data in your enterprise.

The flexibility and extensibility of the SDM ensure that Systinet can keep up with the evolution of your data.

The important concepts of the model are the artifacts types, their properties, and the relationships between artifact types.

Each artifact type represents an abstraction of a particular entity in your organization, for example, a service or organizational unit. The properties of an artifact describe it in business terms and its relationships describe how it fits with the other entities in the Catalog, and by extension, your business.

For example, the following diagram shows artifacts and relationships relating to a service.



For more details about these concepts, see "Artifacts and Properties" on the next page.

For a more detailed description of the use of artifact types and relationships in a service context, see "Application Modeling" on page 28.

Tip: You can customize the SDM to meet your requirements using Customization Editor. For details, see the *Customization Editor Guide*.

For details of the structure of the model, see the SOA Definition Model Documentation available at `http://host:port/soa/web/doc/sdm/index.html`.

Artifacts and Properties

Artifacts are the entities in the Catalog that represent objects in your business. Each artifact is an instance of an artifact type defined in the SDM Model.

For each artifact type, the SDM Model defines a set of properties that describe the artifact, categorize it, and define its relationships with other artifacts.

This document describes artifacts in the following ways:

- **Artifact Instances**

  Artifact instances are entities in the Catalog. Each artifact instance corresponds to a business object, such as a service, an organizational unit, or a contract.

- **Data Artifacts**

  Data artifacts are artifact instances that are associated with specific data content or documents, such as documentation, WSDLs, and XML Schemas.

- **Abstract Artifact Types**

  Abstract artifacts types do not have artifact instances. Their purpose is to define generic artifact types in the SDM Model hierarchy. For example, the SDM Model defines an abstract Implementation artifact type with a set of properties that apply to all implementation artifact types; SOAP Services, XML Services, and Web Applications.

The SDM Model defines a set of properties for each artifact type.

Systinet uses the following property types:

- **Primitive and Complex Properties**

  Primitive and complex properties are attributes of artifact instances. Examples of primitive properties are name, and description. An example of a complex property is address, which consists of further sub-properties; address lines, city, country, etc. These properties describe the artifact instance in terms of the business object it represents.

- **Categorization Properties**

  You can classify artifacts using defined categorization sets. Examples, include geographical location, and business impact. For more details, see "Categorization" on the next page.

- **Relationships**

  Relationships capture logical as well as physical dependencies between artifacts (enabling impact and what-if analysis). Relationship-based properties are either explicitly created by users or implied by the model for a type of source document and related target document. An example of a relationship is the provider relationship between a service and the entity that provides it. The most important concept for relationships is aggregation which enables a single operation on a single artifact to propagate to related artifacts. For details, see "Aggregation" on page 42.

- **System Properties**

  Systinet uses system properties to track information about an artifact such as its revision, access rights, unique identifier, etc. These properties describe the artifact instance in terms of the functionality of Systinet, for example, for security, lifecycle, and artifact history.

For a more detailed description of the use of artifact types and relationships in a service context, see "Application Modeling" on page 28.

Categorization

Categorization provides a way of classifying artifacts according to known and controlled value sets. Artifacts can have properties that are classed as categorization values which can only hold a value from the fixed set of values. Artifacts also use a category bag, enabling them to be classified by a particular categorization even if it is not defined as a property for that artifact type in the SDM model.

Systinet typically uses taxonomies to define sets of fixed values. These are usually referred to as categories. The categories can be organized either hierarchically or as a flat list. For example, the criticality categorization defines a set of values for classifying the business impact of a service failure, with possible values, High, Medium, and Low. Systinet also uses widely recognized categorizations, such Countries (ISO 3166) used to standardize categorization by country or even regions within countries.

Users with the appropriate permissions can categorize artifacts. For details, see "How to Categorize Artifacts" in the *User Guide*.

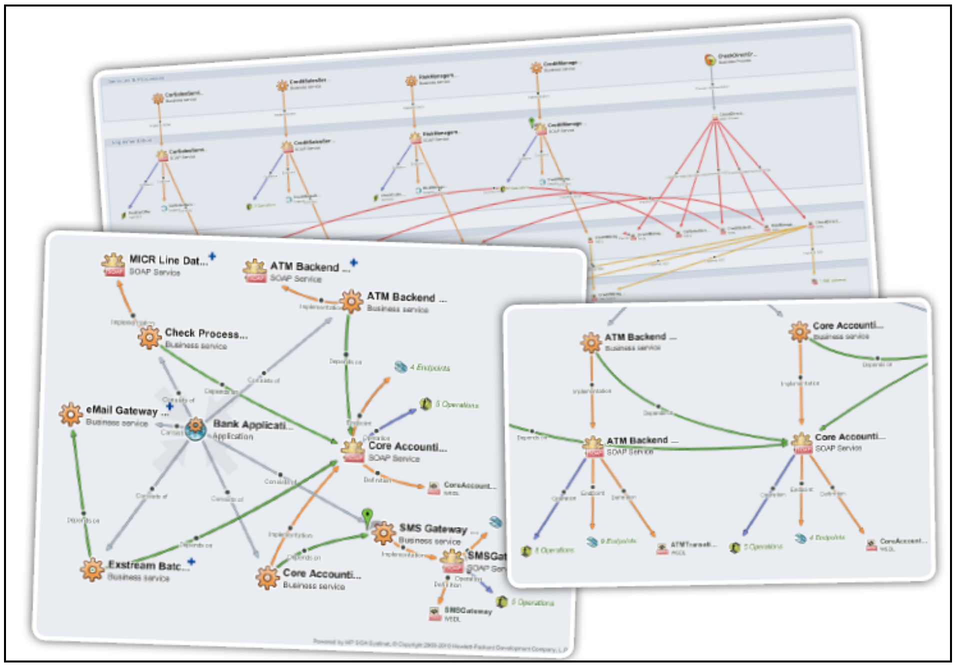
The Catalog tab provides functionality for browsing artifacts by their categorizations and search enables the use of categorization properties. For details, see "How to Browse by Categories" and "How to Search the Catalog" in the *User Guide*.

Lifecycle management can use categories to assign artifacts to different lifecycle processes according to their categorization. For details, see "Automatic Assignment" on page 69.

You can customize the set of categorizations defined in Systinet. For details, see the *Taxonomy Editor Guide*.

Chapter 4: Navigator

The Navigator tab provides visual representation of the content of the Catalog. Instead of focusing on the details of single artifacts, it helps to understand the structure and dependencies of applications, services, and processes in the Catalog. As its name suggests, it is not only an alternative view but also a navigator that enables the exploration, browsing, and traversing of your Catalog content in a graphical way.



The main features of the Graphical Navigator are:

- **Catalog Browsing**

  Simple and rapid navigation of the Catalog content. For details, see "Navigator Tab" in the *User Guide*.

- **Navigator Layouts**

  The navigator enables you to view artifact graphs using different layouts, organizing the artifacts into various structures to demonstrate a particular aspect of their relationships. You can also create customized layouts to focus on particular relationship structures. For details, see "Navigator Layouts" on page 24.

- **Filters**

  The navigator provides a set of role-based filters focusing on the artifacts of interest to each role, as well as the ability to create custom filters. For details, see "Navigator Filters" on page 22.

- **Saved Graphs**

  The graph is interactive, enabling you to modify its appearance by moving, expanding, and collapsing nodes. You can add a particular graph layout to your saved graphs, enabling you to return at a later time to a particular layout for a particular graph. For details, see "How to Use Saved Graphs" in the *User Guide*.

- **Impact and Domain Highlights**

  The navigator enables you to view the impact of change to an artifact within its graph of relations, or to view the artifacts within a particular domain. For details, see "How to Highlight Impact and Dependency" and "How to Highlight Domains" in the *User Guide*.

- **Screenshots**

  Create an image file copy of particular graph layouts. For details, see "Navigator Toolbar" in the *User Guide*.

The navigator can also help collaboration between Systinet users. For example, an architect can demonstrate the structure of an application to the developers who implement it, and the developer can explain dependencies the application has on corporate schemas to QA engineers.

The main benefits of the navigator vary according to the role of the user:

- *Architects* can analyze the structure and dependencies of applications, services, and business processes.

- *Developers* can quickly understand the complex structure of service interfaces defined by interconnected WSDLs and XSDs, and understand the business context of service implementations.

- *QA Engineers* can understand the context and structure of dependencies for tested services and review the impact of any changes in order to focus only on those services affected by a change.

- *Operation Engineers* can assess the purpose and business context of services and applications.

- *Managers* can use the navigator to improve communication with architects and business analysts.

Your application infrastructure can quickly become extremely complex, so it is essential to have the right tools to help you to understand, present, discuss, and optimize your content.

HP Software also provides a set of IDE plugins with integrated functionality that opens the Navigator view within the IDE. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

Example: Banking Application

Consider a Banking Application that consists of the following:

- Six interrelated banking services.

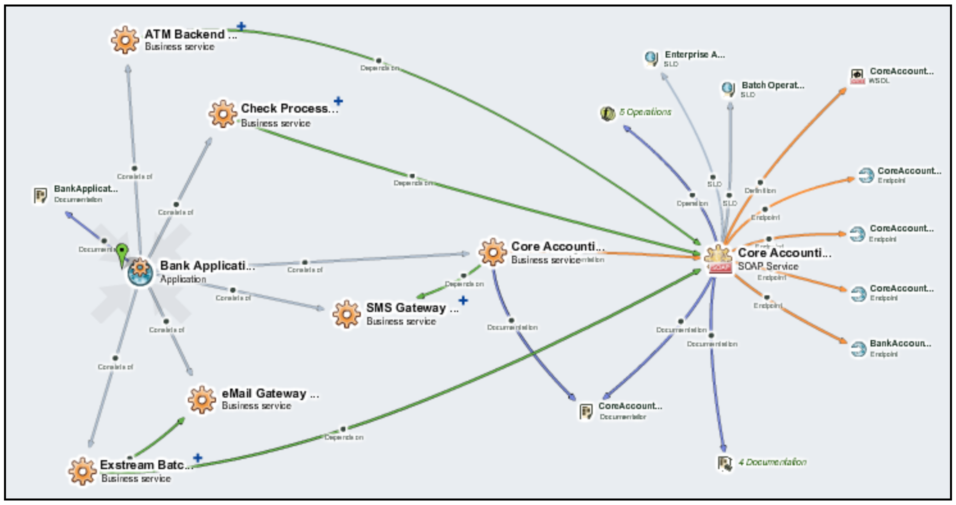
- Application documentation.

Each service can be examined in more depth. For example, the Core Accounting Service consists of the following:

- Three documents about the service.
- A SOAP Service implementation.
- A WSDL defining the implementation.
- Five operations of the implementation defined by the WSDL.
- Four endpoints of the implementation defined by the WSDL.
- Five documents about the implementation.
- Two sets of service level objectives which are applicable to the implementation.

This description is rapidly becoming more and more complex, even though it is only the structure of a single application and one of its services.

The Navigator provides an interactive overview of this complexity in a single image.

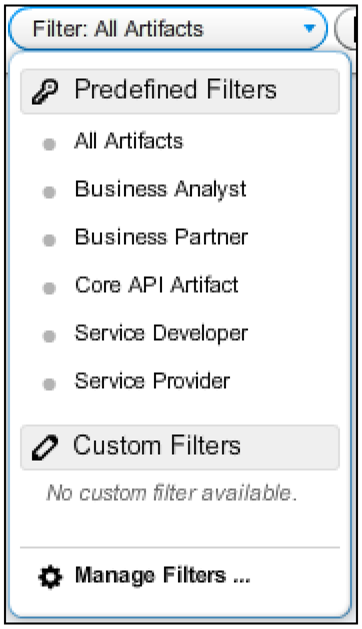


Navigator Filters

The Navigator provides a set of filters enabling you to focus on the artifacts that matter to your role. The filter determines the content of the graph by defining which artifacts display, when to group a set of artifacts, which relationships to show, and what color each relationship uses.

Access Navigator Filters, using the Filters toolbar control.

Screenshot: Navigator Filters Control



Select from the following predefined filters which the administrator can edit:

- **All Artifacts**

  Displays all artifacts that your role has permission to view.

- **Business Analyst**

  Focuses on applications, projects, services, processes, and their implementations, and contract related artifacts.

- **Business Partner**

  Focuses on services and their implementations, and contract related artifacts.

- **Service Developer**

  Focuses on the implementation of services, showing services, their implementations, and service description documents.

- **Service Provider**

  Focuses on the provision of services, showing services, their implementations, endpoints, and operations.

Note: The choice of filters is restricted to those appropriate to your role.

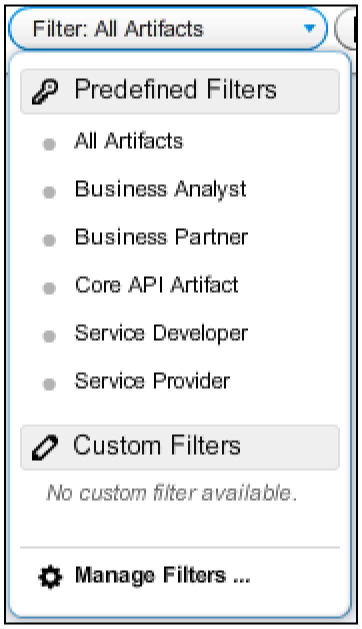
In addition to the predefined filters, you can create your own custom filters. For details about creating, copying, and editing filters, see "How to Manage Navigator Filters" in the *User Guide*.

Navigator Layouts

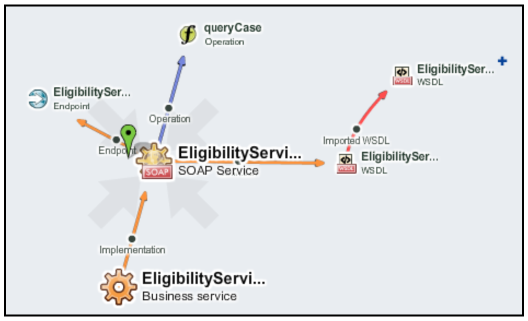
The Navigator offers a number of alternative layouts providing different ways to view Catalog data, enabling you to visualize your data in the way that best suits your needs.

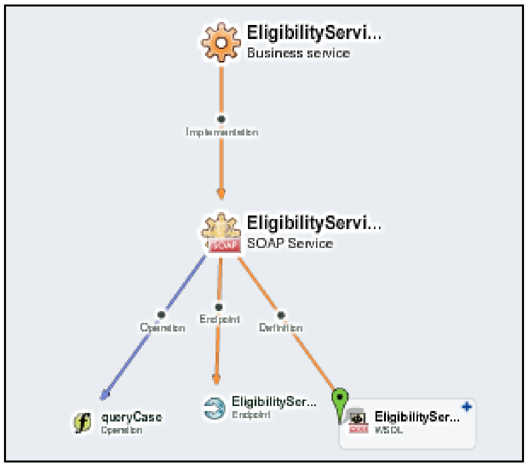
Access Navigator Layouts, using the Layouts toolbar control.

Screenshot: Navigator Layouts Control



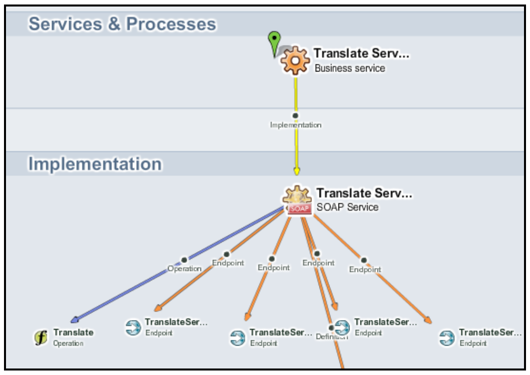
Select from the following predefined layouts:

	**Concentric Radial** The default graph layout places the current artifact in the center of the graph and the related artifacts radiate outwards in all directions. This layout enables you to explore the immediate neighborhood of an artifact.



Hierarchical Layout
This layout organizes the graph into a vertical artifact hierarchy according to the artifact relationships and any defined artifact aggregation structures.

The administrator can configure aggregations and there are a set of default aggregations in Systinet. For details about Aggregation, see "Aggregation" on page 42.



Layered Layouts

Layered layouts organize the hierarchy of artifacts into identified layers, enabling you to focus on important abstractions and analyze dependencies between layers. Each expandable layer can consist of multiple artifact types with relationships within a layer and between them.

Systinet provides the following default Layered Layouts:

- **Architectural Layers** provides a holistic view of your SOA architecture. It divides artifacts into Application, Process, Service, and Infrastructure Layers.

- **Implementation Layers** renders the internal architecture of services and their implementations. It divides artifacts into Services & Process, Implementation, WSDLs, and XML Schemas layers.

- **Project View** provides a view tailored to projects and their content. It enables you to analyze the dependencies between projects, their content, and used data types.

Systinet enables you to create your own customized layered layouts. For details about managing custom layouts, see "How to Manage Navigator Layouts" in the *User Guide*.

In addition to the various layout schemes, you can highlight the impacts between artifacts and artifacts in specific domains. For details, see "How to Highlight Impact and Dependency" and "How to Highlight Domains" in the *User Guide*.

Chapter 5: Collaboration

Systinet is a collaborative platform enabling you to actively participate and provide information to other users throughout the service lifecycle.

Systinet provides the following collaboration features:

- **Comments**

  The Artifact Details page provides a Discussion area where you can start threads and respond to other comments. For details, see "How to Use Comments" in the *User Guide*.

- **Notifications**

  You can send e-mail notifications about an artifact to specified stakeholders. For details, see "How to Use Notifications" in the *User Guide*.

- **Events**

  Events keep you up-to-date with changes to artifacts that you are a stakeholder in and user actions that may impact you. For details, see "How to Use Events" in the *User Guide*.

- **Tasks**

  Tasks provide you with updates for lifecycle and contract tasks assigned to you. For details, see "How to Use Tasks" in the *User Guide*.

- **Ratings**

  You can apply an individual rating to an artifact, contributing to an overall rating that enables other users to select the best artifacts. For details, see "How to Use Ratings" in the *User Guide*.

- **Feeds**

  Systinet provides artifact and search feeds that enable you to track changes and discussions about artifacts in your feed readers and in the Reports tab. For details, see "How to Use Feeds" in the *User Guide*.

- **Sharing**

  Systinet provides a simple action to provide all users with read access to an artifact, making it visible across the whole user-base. For details, see "How to Share Artifacts" in the *User Guide*.

Systinet is also a collaborative platform in combination with your development environments. HP Software provides plug-ins for widely used IDEs that enable your developers to collaborate using the discovery and publishing features of Systinet directly from their IDE. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

Chapter 6: Authoring

Authoring is the process of adding content to the Catalog for the purposes of governance and management.

Authoring consists of the following overlapping areas of functionality:

- Manually creating artifacts to represent entities that you want to develop, govern, and publish.
- Manually enhancing these artifacts with additional properties that represent important information about these entities.
- Manually organizing the content by creating relationships between artifacts.
- Publishing data artifacts with attached content. The publishing process may automate some of the manual authoring steps.

This topic describes the following authoring concepts:

- "Application Modeling" on the next page

  Different organizations model their application infrastructure in different ways. This topic provides a concept reference to the various *service artifacts* in Systinet and ways to use them to meet your requirements.

- "Environments" on page 34

  Most organizations deploy services and applications to multiple environments during their lifecycles. This topic describes the concept and implementation of environments in Systinet.

- "Publishing" on page 35

  Systinet provides extensive support for publishing service definition documents, such as WSDLs. This topic describes the various options available for streamlining your content publishing.

- "Service Provision Workflow" on page 38

  This topic describes the basic steps involved in taking a service from a design document, to a service in production and ready for consumers to use.

The concepts described above involve using the following authoring procedures described in the *User Guide*:

- "How to Create Artifacts"
- "How to Manage Contacts"

- "How to Manage Relationships"
- "How to Attach Documentation"
- "How to Use Projects"
- "How to Publish Content"
- "How to Browse WSDLs and XML Schemas"

In addition to these authoring processes, "Artifact Management" on page 41 describes additional processes that you may need as part of your overall authoring process.

HP Software also provides a set of IDE plugins with integrated publishing functionality. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*

Application Modeling

Different organizations model their application infrastructure in different ways. The open architectural model of Systinet enables you to choose how you create, govern, and provide your application infrastructure in a way that matches your needs.

There are a number of different definitions for *service artifacts* that recognize the various ways that different organizations describe their service infrastructure. Systinet provides the following artifact types with suggestions of how you may want to use them:

Service

A service is a set of functionality that you can provide or use as a single entity. The purpose of the service artifact is to provide an abstraction in Systinet that you can use as a central point of governance, provision, and use that describes the functionality and the users, groups, and contacts responsible for its maintenance and support. For more details, see "Service Artifacts" on page 30.

Business Process

A business process is a set of functionality that you can provide or use as a single entity. The purpose of the process artifact is to provide an abstraction in Systinet that you can use as a central point of governance, provision, and use that describes the functionality and the users, groups, and contacts responsible for its maintenance and support. For more details, see "Business Process Artifacts" on page 30.

Application Interface

An implementation represents the technical and functional aspects of a service or business process. Each service or process may have a number of implementations to different endpoints or with different versions. The implementation artifact provides a central reference point for all these technical and functional aspects of the service or process. An implementation typically consists of an endpoint that the implementation is deployed to, the operations that the implementation performs, and a set of definitions documents that describe the service in technical terms. The elements of an implementation are described in more detail in "Implementations" on page 31.

Application

An application typically provides a set of services and processes. The application artifact enables you to collect a set of services and processes together to govern and provide them as a single entity. Use the generic add relationship functionality to add artifacts to an application. For details, see "How to Manage Relationships" in the *User Guide*.

Project

A project typically contains a set of artifacts with a shared development lifecycle. The project artifact enables you to collect these artifacts together and govern them as a single entity. Use the generic add relationship functionality to add content to a project or use specific project functionality. For details, see "How to Manage Relationships" and "How to Use Projects" in the *User Guide*.

Using these artifact types and the relationships between them provides a platform to manage, govern, and provide your content at the most appropriate level for your organization. For more details about relationships, see "Relationships" on page 32.

An important concept for artifact management is aggregation. Aggregation enables you to perform actions on one artifact which then propagate to all the related sub-artifacts. For more details, see "Aggregation" on page 42.

Consider the following factors when you choose the most appropriate artifact types to govern and provide your infrastructure:

Provision and Use

For the purpose of forming contracts, the model contains an Service Level Objectives artifact type to represent the terms that you offer for a piece of functionality. The default functionality of Systinet enables an SLO to be associated with Applications, Services, Implementations, Endpoints, and Operations.

How you want to offer your business functionality to consumers may determine your choices when you create, govern, and provide your service infrastructure. For more details, see "Contracts" on page 50.

Lifecycle Governance

For the purpose of governing a lifecycle, Systinet provides default lifecycle processes for Applications, Services, Implementations, Projects, Processes, and Process Implementations.

How you want to govern the lifecycle of your business functionality may determine your choices when you create, govern, and provide your service infrastructure. For more details, see "Lifecycle" on page 61.

Versioning

During their lifetime services and processes evolve, so there may be different versions that require management. Systinet supports various versioning strategies enabling you to introduce new versions in a controlled way, which you can then manage, govern, and provide separately.

How you want to support versioning may determine your choices when you create, govern, and provide your service infrastructure. For more details, see "Versioning" on page 47.

Service Artifacts

Artifacts in Systinet represent an abstraction of your services enabling you to visualize and manage your service development, governance, and provision.

The service artifact is usually the central element of a set of artifacts representing a particular set of functionality that you provide or use.

In practice, a service cannot be represented by a single artifact and consists of a set of the following related artifacts:

- **Service**

  The key artifact representing the concept of the service. This artifact contains properties identifying the owners, contacts, and details about the set of functionality that it represents. Other details are represented in related artifacts.

- **Implementation**

  Services typically have at least one implementation. Each implementation represents functional elements of the service. There are a number of different kinds of implementation and they relate to sets of technical artifacts representing the operations the implementation performs, the endpoints representing the servers or environments where the implementation is available, and definition documents such as WSDLs and XSDs that describe the service and its implementation. For more details, see "Implementations" on the next page.

- **Documentation**

  Services are often described by different documents such as specifications and user guides.

- **Service Level Objectives**

  Providers of services can offer sets of terms to service consumers describing the service levels that they expect to meet. When a consumer requests the use of a service, the service level objective defines the terms of the contract between the provider and the consumer. For details about contracts, see "Contracts" on page 50.

Systinet enables associates these different artifacts together as a single service aggregate enabling you to manage, govern, or use them all as a single object. For more details about aggregates, see "Aggregation" on page 42.

Business Process Artifacts

Systinet represents an abstraction of your processes enabling you to visualize and manage your process deployment and governance.

The Business Process Artifact is usually the central point of a process infrastructure representing a particular set of functionality that you provide or use.

In practice, a business process cannot be represented by a single artifact and consists of a set of the following related artifacts:

- **Business Process**

  The key artifact representing the concept of the process. This artifact contains properties identifying the owners, contacts, and details about the set of functionality that it represents.

- **Process Implementation**

  The entity representing the process implementation. A Business Process typically has one BPEL Process Implementation, which refers to related (consumed and provided) services and their implementations using an aggregation of WSDL documents. BPEL Process implementation is typically accessible and controlled by one service implementation, so a process implementation is also visible as a service implementation, and can be reused by another service or process. For details, see "Implementations" below.

- **Documentation**

  Processes are often described by different documents such as specifications and user guides.

Systinet associates these different artifacts together as a single process aggregate enabling you to manage, govern, or use them all as a single object. For more details about aggregates, see "Aggregation" on page 42.

Implementations

Implementations represent the technical and functional elements of a service or process. Systinet supports the following implementation artifact types:

- **Application Interfaces**
  - SOAP Service
  - XML Service
  - REST Service
  - File Transfer
- In the earlier version of the guide Application Interfaces was called as Service Implementations.
- **Process Implementations**
  - BPEL Process
  - XPDL Process

Implementations are typically associated with the following artifact types:

- **Endpoints**

  Endpoints represent URLs used to access a service or process implementation. Throughout the lifecycle of a service or process, there are likely to be a number of different environments hosting the service or process, for example, development, testing, and production servers. Use a different endpoint to represent each of these environment implementations.

- **Operations**

  Operations represent a method defined in an implementation.

- **WSDLs**, **XML Schemas**, **BPELs**, and **XPDLs**

  Services, processes and their implementations are often described by definition documents such as WSDLs or BPELs. Systinet provides publishing functionality which decomposes definition documents, analyzes the service infrastructure it defines, and then creates or updates the relevant artifacts in the Catalog. For more details, see "Publishing" on page 35.

  Note: BPEL and XPDL definition documents are directly attached to the relevant BPEL and XPDL Process artifacts.

  Note: After the publication of a BPEL/XPDL file, there must be a "consists of" relationship from the BPEL/XPDL process artifacts to the SOAP Service artifacts which define the imported WSDLs of BPEI/XPDL.

Implementations and Endpoints can use different messaging transport methods. Systinet supports HTTP, HTTPS, SMTP, and JMS transport methods.

Typically, implementations are deployed to several endpoints during their lifecycle which may represent different environments. Systinet provides environment support which reduces the overhead involved with maintaining artifacts for multi-environment deployment. For details, see "Environments" on page 34.

Relationships

The relationships between artifacts represent one of the most significant aspects of authoring and artifact management.

For each artifact type, the SDM Model defines a set of relationships where each named relationship defines a link between that artifact type and another artifact type which is expected to be associated with it. For details about the model, see "Service Definition Model (SDM)" on page 17.

Every relationship type is directional from one artifact type to another and has an inverse relationship in the opposite direction. For example, the relationship from a Service Artifact to a Documentation Artifact is the *Documentation* relationship and the inverse relationship from the Documentation Artifact to the Service Artifact is the *DocumentationOf* relationship.

Relationships can be broadly divided into the following types:

- **Dependencies**

  A *Dependency* is a directed relationship indicating that an artifact uses another artifact. For example, Service A may have a *depends on* relationship to Service B to indicate that Service A uses Service B and that any change to the Service B should take into account the impact on Service A. Contract relationships are a specific example of dependency relationships indicating the contractual use of an artifact according to specified usage terms. Various functions in the UI offer an **Include Dependencies** option (for example, Export to ZIP Archive) which takes these dependencies into account when performing actions on an artifact.

  The inverse of a dependency is an *Impact*. Impact enables you to view the dependent artifacts which may be affected by a change to an artifact. The most useful way to view impact and dependencies is in the Navigator view. For details, see "How to Highlight Impact and Dependency" in the *User Guide*.

- **Aggregations**

  Aggregation relationships enable a set of related artifacts to be represented as a single abstraction in order to perform operations on the entire set. For example, a service aggregation consists of not only the service itself, but also its implementations, operations, endpoints, SLOs, documentation, etc. The concept of aggregation has a significant impact on the functionality of Systinet and is discussed in more detail in "Aggregation" on page 42.

- **Unclassified Associations**

  In addition to relationships in the public model, there are additional system relationships which Systinet automatically applies in specific circumstances. For example, when you create a new version of an artifact, a previous version relationship links the new version to the old one. This relationship enables you to notify users of the old version when the new version is ready to use.

The user interface provides extensive support for named relationship types, showing the most important relationship in the Artifact Details page Overview tab, and all the expected relationships in the Details tab. The Relationships tab provides an alternative view with the relationships organized into Outgoing and Incoming Relationships. For details, see "Artifact Details Page Overview Tab", "Artifact Details Page Details Tab", and "Artifact Details Page Relationships Tab" in the *User Guide*.

In addition to the relationships defined by the model, there are use cases where a more generic relationship may be useful. Examples include defining project content, where the content may be any artifact type, or composite services where a particular service may be made up of other services. To support these use cases, Systinet includes generic **Consists Of** and **Depends On** relationships, representing sub-artifacts and parent artifacts respectively. Use these relationships to associate artifact types together when a specific named relationship does not exist.

Manage both named and generic relationships from the Artifact Details page. For details, see "How to Manage Relationships" in the *User Guide*.

Environments

Most organizations use several environments to clearly separate production from testing and development. This means that when a service moves through its lifecycle, it is likely to be deployed multiple times in a number of different environments.

Systinet makes use of an Environment categorization to represent these environments. Each environment is represented by an environment category value which can be applied to endpoints and also to server artifacts for other products integrated with Systinet (for example, UDDI and BSM servers).

This categorization ensures that only the appropriate information is automatically exchanged between Systinet and integrated servers. For more details about integrated servers, see "Product Integration" on page 71.

This approach avoids the constant republishing of WSDLs defining services to reflect these additional endpoints as the service moves through its lifecycle. Systinet provides environment specific WSDLs that automatically contain the appropriate endpoints whenever a WSDL is required.

A typical example for a service could be described as follows:

- **Development**

  A service is likely to be defined by publishing a WSDL before it reaches the development stage. When it reaches the development stage it requires an endpoint representing a development server. Instead of amending and republishing the WSDL, Systinet enables you to add an endpoint and categorize it as `Development`.

- **Testing**

  During the testing stage, the service may require an additional endpoint representing test deployment of the service and this is also achieved by adding a new endpoint categorized as `Testing`.

  Systinet may be integrated with *Application Lifecycle Manager* (ALM). When a service is registered in ALM, the associated WSDL for the test environment is exported to ALM. By categorizing the ALM server as `Testing`, you ensure that only the Testing endpoints are visible to the ALM server. For more details about ALM integration, see "ALM Integration" on page 77.

- **Staging**

  A staging environment is intended to duplicate the production environment as closely as possible. Staging environments are often used for final pre-production testing, or for demonstration and training purposes. A service in a staging environment requires an additional endpoint representing the staging deployment categorized as `Staging`.

- **Production**

Similarly in production, the service requires an additional endpoint representing the production deployment of the service and this is also achieved by adding a new endpoint categorized as `Production`.

When a service reaches production you may wish to export it to a UDDI registry. By categorizing the registry server as `Production` you ensure that the service is exported with the appropriate production endpoints. For more details about UDDI Registry introduction, see "UDDI Registry Integration" on page 75.

Publishing

Files can be uploaded to Systinet with the Import File wizard accessible from the Import menu in the Catalog tab, or as Upload actions in Artifact Detail page Details tabs.

Systinet does not change uploaded files, they are stored and available as they were uploaded. You can upload local files or remote files exposed at a URL. Moreover, you can upload a ZIP archive containing multiple files. As well as the file to upload, you can select the server folder and set additional options.

Import File functionality offers the following options:

- **Server Folder**

  The server folder determines the path in the repository workspace where Systinet stores uploaded files.

  The correct use of server folders is crucial for definition data management. As mentioned above, definition data files usually refer to each other via relative links and so it is important to set server folders so that relative links work. For more details and examples, see "Server Folders" on page 37.

  Each uploaded file is also accessible using the REST location space in the following URL format, `http://hostname:port/systinet/platform/rest/location/serverFolder/fileName`. For example, if you upload `a.wsdl` to server folder `/accounting` then the WSDL document is accessible using `http://hostname:port/systinet/platform/rest/location/accounting/a.wsdl`. For more details, see "Publishing Location Feeds" in the *Developer Guide*.

- **Authentication**

  This option applies when you upload remote documents and the remote server requires credentials. Import provides the option to input credentials and to store them for future use. For details, see "How to Manage Your Credentials" in the *User Guide*.

- **Synchronization**

  This option applies when you upload remote documents. Systinet provides the option to synchronize the content of the repository with remote files. This feature is important when Systinet is not the master store for definition files. The administrator can schedule a synchronization task that synchronizes all documents in Systinet with remote master files.

The behavior of synchronization depends on the following options:

  - **Automatic**

    Automatically update the repository file if the remote file is updated.

  - **Manual**

    If the remote file is updated, the repository file is marked as *out-of-sync*. Systinet provides functionality to update each out-of-sync repsoitory file manually.

  - **Disabled**

    No synchronization takes place.

- **Process Archive Content**

  This option applies when you upload a ZIP archive. If selected (default), Systinet extracts the archive and uploads all the content. Otherwise, Systinet uploads the archive as documentation content attached to a documentation artifact. ZIP, JAR, WAR, EAR, and BPR archives are supported by default, but you can extend the predefined set of supported archives.

- **BPEL Decomposition**

  This option applies when you upload BPEL or SCA documents. It is similar to WSDL decomposition. You can select whether to create only data content artifacts, such as SCA or BPEL artifacts (None option) or whether to create a Business Process for each BPEL artifact.

- **WSDL Decomposition**

  This option applies when you upload a WSDL. You can select from the following WSDL decomposition options to enable the publishing process to create the artifacts defined by the WSDL:

  - **None**

    Create only data content artifacts, such as WSDL and XSD artifacts, with attached data content.

  - **Implementations**

    Create data content artifacts and also parse the WSDLs and create appropriate implementation artifacts, such as SOAP Service, Operation, and Endpoint artifacts.

- **Services**

  In addition to the implementation artifacts, create appropriate service artifacts.

- **Service Type**

  This option applies if you upload a WSDL document and create services when WSDL decomposition is set to Services. You can specify the service type (infrastructure, business, or application service) for the created service.

Note: Systinet controls the functionality of publishing with a set of configurable properties. For details of these properties and how to change them see the Publishing table in "System Configuration Properties" and "How to Manage System Settings" in the *Administration Guide*.

For more details about the process of publishing, see "How to Publish Content" in the *User Guide*.

Publishing includes support for versioning. For details about using publishing to control versioning, see "How to Upload Versioned Data Files" in the *User Guide*.

Server Folders

Server folders specify data file locations in the Systinet repository workspace.

Systinet offers the repository workspace as a primary storage location for data definition files. By specifying server folder you can determine the storage location in the workspace which is crucial for retaining valid relative links to already uploaded documents. The workspace can be browsed with the select server folder dialog in the File Import page or with the REST Interface.

HP recommends using relative links and to design server folders in the workspace appropriately. For more details, see "Definition Data Best Practice" in the *User Guide*.

Systinet exposes the workspace using REST as a public interface. You can browse the workspace at `http://hostname:port/systinet/platform/rest/location`. This URL opens a list of top level server folders and files uploaded directly to the root folder.

You can download a selected folder as an archive. For example, you can download all data definition files in a server folder to your local disk as a ZIP archive, change them according to your needs and then upload them again to the repository. Alternatively, you can use Systinet IDE Integration Plug-Ins or WebDAV for this purpose. For more details, see *Plugin for Eclipse*, *Plugin for Visual Studio*, and "WebDAV Compliant Publishing" in the *Developer Guide*.

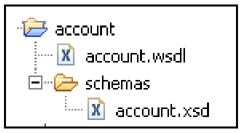
Note: Systinet does not support the upload of ZIP files using WebDav. If you need to upload a ZIP file, use the import functionality of the UI or upload functionality of the IDE Plug-Ins.

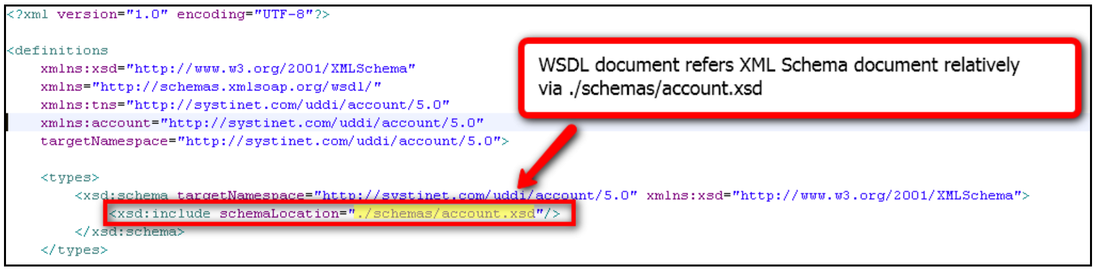
You can browse a selected folder by clicking its name. When you view the content of specific folder, the URL changes. For example, when you browse the `/accounting` folder then the URL is `http://hostname:port/systinet/platform/rest/location/accounting`. For more details, see "Publishing Location Feeds" in the *Developer Guide*.

The following examples describe WSDL and XML schema documents, but are also valid for all data definition files with relative references:

Uploading a WSDL Document and Creating a New XML Schema

In this example you upload a ZIP archive including `account.wsdl` and `account.xsd`. The WSDL document contains relative reference to `account.xsd`. The following diagrams describe the relative reference and the ZIP structure. Notice that the relative links correspond to ZIP structure.





```
<?xml version="1.0" encoding="UTF-8"?>

<definitions
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:tns="http://systinet.com/uddi/account/5.0"
    xmlns:account="http://systinet.com/uddi/account/5.0"
    targetNamespace="http://systinet.com/uddi/account/5.0">

    <types>
        <xsd:schema targetNamespace="http://systinet.com/uddi/account/5.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <xsd:include schemaLocation="./schemas/account.xsd"/>
        </xsd:schema>
    </types>
```

WSDL document refers XML Schema document relatively via ./schemas/account.xsd

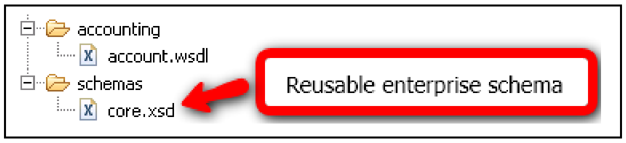
Both files are new and do not exist in the repository. When you upload, specify server folder `/accounting` and so Systinet creates `account.wsdl` in `/accounting/account.wsdl` (accessible from `http://hostname:port/systinet/platform/rest/location/accounting/account.wsdl`) and `account.xsd` in `/accounting/schemas/account.xsd` (accessible from `http://hostname:port/systinet/platform/rest/location/accounting/schemas/account.xsd`).

Uploading a WSDL Document and Reusing an Existing XML Schema

When you want to reuse an existing XML Schema document you must use the correct relative link in your WSDL document and specify an appropriate server folder when uploading.

Assuming that you want to reuse an existing `core.xsd` in a shared `/schemas` folder in a new `account.wsdl` that you upload to the `/accounting` server folder. You need to use an appropriate relative link in your WSDL document.

Examining the intended structure of the repository workspace after uploading `account.wsdl`, the link must be `./../schemas/core.xsd`.

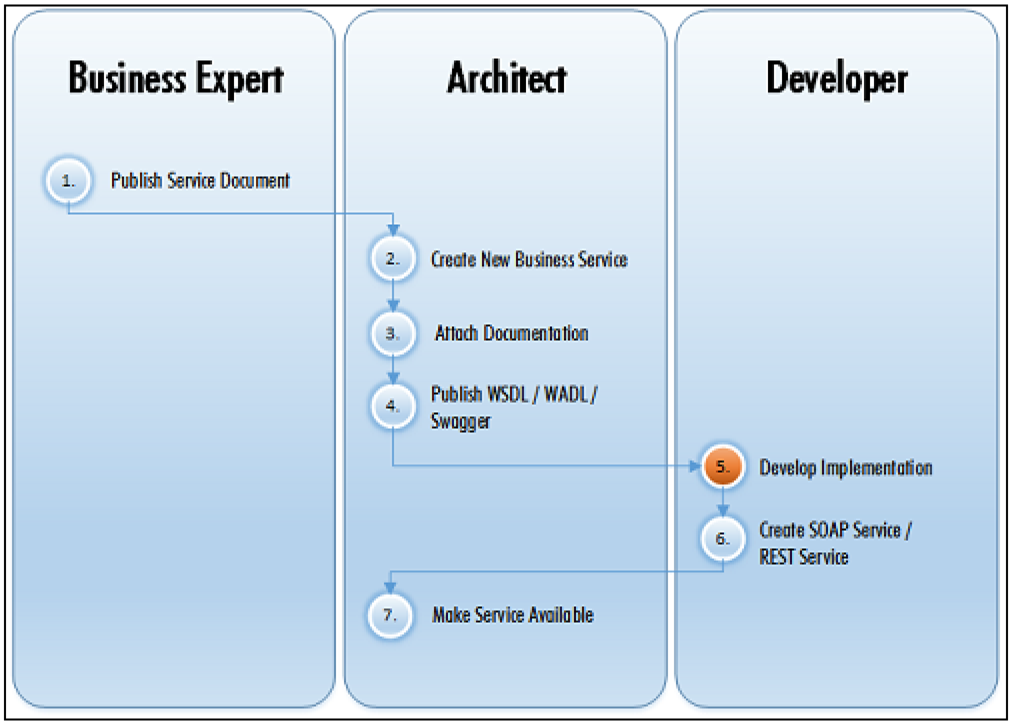


Service Provision Workflow

Systinet uses a set of authoring processes, enabling you to add services to the Catalog and make them available to consumers.

Note: This workflow considers the steps required for Service Provision separately from Lifecycle Management. In practice, these steps are associated with particular lifecycle stages with relevant verification and approval processes. For more details, see "Lifecycle" on page 61.

The diagram describes a typical service provision process and the steps required to achieve it. The workflow is split into the user roles that would typically perform the actions in an organization.


Business Expert
Architect
Developer
1. Publish Service Document
2. Create New Business Service
3. Attach Documentation
4. Publish WSDL / WADL / Swagger
5. Develop Implementation
6. Create SOAP Service / REST Service
7. Make Service Available

1. The Business Analyst publishes a service document in Systinet describing a proposed new service. For details, see "How toPublish Content" in the *User Guide*.

2. The Architect receives notification of the new service requirement, reviews the documentation, and creates a new service artifact. For details, see "How to Create Artifacts" in the *User Guide*.

3. The Architect creates a new technical document and attaches the new document and the existing service document to the service. For details, see "How to Attach Documentation" in the *User Guide*.

4. The Architect publishes a WSDL document, which contains a technical description of the service. The architect can use the publishing functionality of Systinet or develop the WSDL in an integrated IDE and publish it directly. For details, see "How to Publish Content" in the *User Guide*.

5. The Developer reviews the documentation and uses the WSDL to develop an implementation of the service. Typically, the developer would use an integrated IDE to develop the implementation. HP Software provides plugins for Eclipse and Visual Studio that integrate Systinet discovery and publishing functionality in these IDEs. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

6. The Developer publishes the WSDL. Systinet publishing functionality creates the artifact structure defined by the WSDL and modifies any relevant existing artifacts. The Developer can use the publishing functionality in Systinet or publish directly from an integrated IDE. For details, see "How to Publish Content" in the *User Guide* or *Plugin for Eclipse* and *Plugin for Visual Studio*.

7. The Architect edits the service and makes it available for consumption. For details, see "How to Edit Artifact Properties" in the *User Guide*.

Chapter 7: Artifact Management

When you have data in the Catalog, there is a requirement to maintain and update it. Systinet provides extensive support for artifact maintenance covering the following functionality described in the User Guide:

- "How to Use Bulk Operations"

  Maintaining artifacts individually is time-consuming, so Systinet offers bulk operations, enabling you to maintain multiple artifacts in a single operation.

- "How to Edit Artifacts"

  Artifacts contain a lot of information, so Systinet offers numerous edit options, enabling you to focus on particular aspects of the artifacts that you want to edit.

- "How to Delete Artifacts"

  Artifact deletion is a two-step process. Deleting artifacts places them in the Recycle Bin where they can then be restored or permanently removed from the Catalog. For detail, see "Recycle Bin" on page 45.

- "How to Synchronize Artifacts"

  Artifacts in the Catalog may have attached data content which originated from external content. Systinet provides change management with a synchronization tool enabling you to keep track of changes to the external content and to manually or automatically update the version of that content stored in the repository.

- "How to Export and Import Artifacts"

  Your deployment may be deployed across several instances of Systinet. Export and import functionality enables you to transfer content across instances or maintain backed-up versions of content for later reference. For more details about multi-instance deployment, see "Federation" on page 43.

- "Definition Data Best Practice"

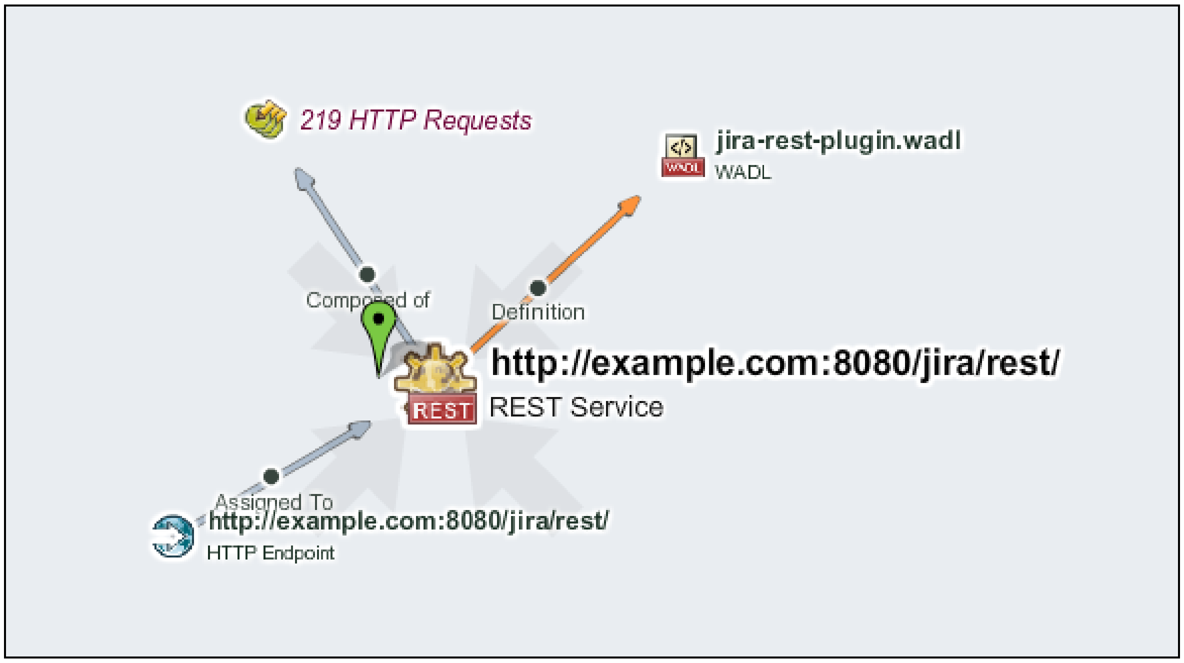
  Systinet provides extensive support for definition files, such as WSDLs and XML Schemas. This best practice topic describes the recommended ways to exploit this functionality to best effect.

Note: Systinet controls access to artifact management functionality based on user roles and permissions. For details, see "Security and Access Control" on page 90.

Aggregation

Abstractions such as services are not represented in the Catalog by a single artifact. Artifacts in Systinet form natural hierarchies corresponding to hierarchies of business objects. It is necessary to be able to maintain, govern, and provide these hierarchies or *aggregates* as a whole. For example, when moving service to a different domain it is necessary to be able to move not only to the service artifact itself but also to its aggregated artifacts such as SLOs, documentation, implementations, and endpoints.

For example, the following diagram shows a service aggregate with an implementation, its operations, endpoint, and WSDL. Performing a bulk operation on the service such as delete, offers the **Include Sub-artifacts** option, which would also delete the implementation, operations, endpoint, and WSDL.



Systinet constructs these hierarchies or *aggregates* as follows:

- **Aggregations** are relationships whose relationship type is marked as an aggregation (relationship types in the model can be marked as aggregations, dependencies, or generic relationships). For example, the relationship *implementation of* between Service and Implementation artifact types is marked as an aggregation. Whenever there is an aggregation between artifacts A and B, we say that A aggregates B, for example, a service aggregates its implementations.

- **Sub-artifacts** of an artifact are all the artifacts that the artifact directly or indirectly aggregates (indirectly aggregated artifacts are artifacts aggregated by the artifact or any of its recursively aggregated artifacts).

- **Aggregates** are an artifact together with all its sub-artifacts. For example, an implementation with all its endpoints, operations, and WSDLs. Many operations, for example, change owner, can be applied to whole aggregates by selecting the **Include Sub-artifacts** option.

You can maintain, govern, and provide *aggregates* using the following functionality:

- **Artifact Export**

  When you export an artifact, Systinet also exports all related sub-artifacts in the same aggregate. For details, see "How to Export and Import Artifacts" in the *User Guide*.

- **Artifact Operations**

  Most operations that you can perform on an artifact or set of artifacts, such as delete and edit, offer an Advanced Option to **Include Subartifacts**. Selecting this option propagates the same operation to all sub-artifacts in the aggregate.

Another function that uses the ability to classify relationship types as aggregations or dependencies (or just regular relationships) is *Impact Analysis*. It enables you to assess the potential effect of a change. When assessing the impact of a change for an artifact, aggregations and dependencies are considered. So, artifacts that aggregate or depend on an artifact (directly or indirectly) may be impacted by a change.

You can visualize these aggregate structures and the impacts and dependencies they form using the following features:

- **Graphical Navigator**

  The Graphical Navigator provides a visual representation of the Catalog content. You can use hierarchical and custom layered layouts to examine specific aggregate structures and use highlighting to focus on the impact and dependencies between artifacts. For details, see "Navigator" on page 20.

- **Artifact Detail Tree View**

  The Tree View tab in Artifact Details pages enables you to view and navigate through the aggregates that the artifact is part of. You can switch between Impact view listing parent artifacts and Dependency view listing sub-artifacts. For details, see in the *User Guide*.

Federation

Whether because of regulatory (for example, ISO 17799-based Separation of Environments and Duties) or organizational reasons, it may be necessary to deploy multiple physically separated repositories instead of a single centrally managed repository. In such environments, the synchronization and maintenance of data can prove problematic if the repository does not address these problems directly.

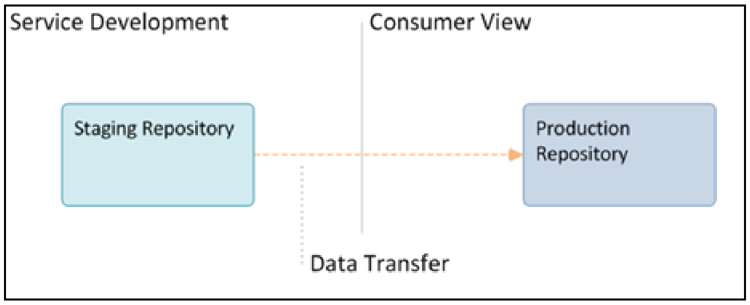
Systinet introduces new features that help to federate physically separated repositories in a controlled fashion. Users can control which data to federate by selecting a domain, project, or even particular applications that are subject to migration between repositories. They can also control whether lifecycle metadata, security, settings, etc. migrate together with the data or are adopted from the target repository. Systinet supports various federation topologies – for example, ‘department’ repositories federated in a central enterprise repository or an ‘air-gap’ scenario where there is no direct connectivity between the repositories.

Systinet provides the following scenarios, functionality, and concepts to support federation:

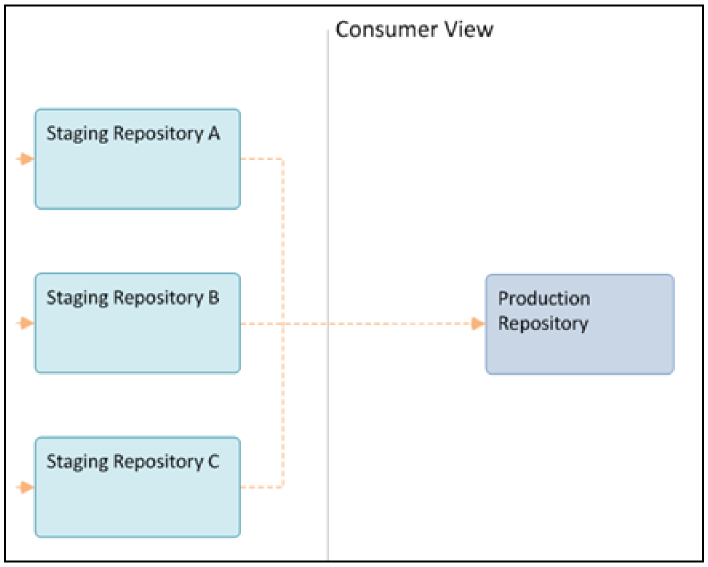
Multiple instances of Systinet

Many organizations require a physical separation between pre-production and production environments.

A typical use case for this scenario is a Staging instance for pre-production to manage the lifecycle of a development project, and to establish all the supporting data for services which can then be transferred to a Production instance which is primarily used by consumers and captures contract information.



An alternative is multiple Staging instances for separated development paths.



Domains

Systinet provides domain support enabling you to separate the Catalog into distinct areas. These domains are available for use in the same way as separate instances of Systinet. For more details, see "Domains" on page 92.

Environments

Systinet enables you to apply specific environment categorizations to different endpoints or integrated product servers. This categorization enables the use of a single service definition document to define the same service across different instances or environments. For more details, see "Environments" on page 34.

Artifact Export / Import

Supporting the transfer of data between separate instances or domains is a selective export / import process. Export utilizes the concept of aggregation enabling you to perform an export on a single artifact, which collects all the related sub-artifacts in a single archive which can then be imported elsewhere. A typical scenario is to export a project from a Staging instance to a Production instance when its development lifecycle is complete and it is ready for use by consumers. For details about aggregation, projects, and export / import, see "Aggregation" on page 42, and "How to Use Projects", and "How to Export and Import Artifacts" in the *User Guide*.

Lifecycle and Domain Export / Import

In addition to transferring artifacts, Systinet uses the same export / import functionality to support the transfer of lifecycle processes and domains. Exporting a lifecycle process enables you to take advantage of generic lifecycle templates featuring the use of roles instead of specified users and groups and apply the same process across multiple instances or domains. Domain export enables to transfer of the entire content of a domain, including its configuration to another instance or domain. For details, see "How to Export Lifecycle Processes" and "How to Export Domain Content" in the *Administration Guide*.

Repository Export / Import

In addition to the export / import functionality provided in the user interface for artifact, lifecycle, and domain transfer. The administrator has command-line export / import tools to create backups or copies of the entire repository including all its configuration and security data. For details, see "Export Tool" and "Import Tool" in the *Administration Guide*.

Recycle Bin

Systinet places all deleted items in the Recycle Bin.

Note: By default, Systinet runs a recycle bin cleaner task every hour which automatically purges artifacts deleted more than 30 days ago. To change this default, change the **platform.recycleBin.timeout** property (listed in seconds) using the functionality described in "How to Manage System Settings" in the *Administration Guide*. To change the task settings, see "Administration Task Management" in the *Administration Guide*.

You can access the Recycle Bin from the Catalog Tab View menu.

The Recycle Bin displays a table of deleted artifacts with the following options:

- **Delete Permanently**

  Select artifacts and then click **Delete Permanently** to purge them from the repository.

- **Restore**

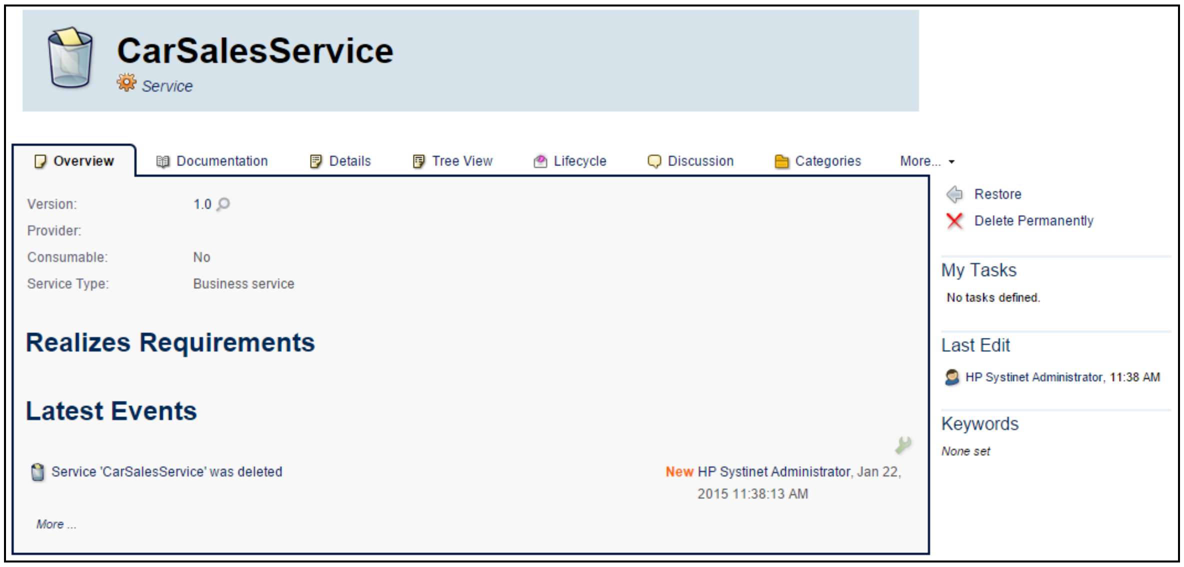
  Select artifacts and then click **Restore** to make them available again in the repository.

- **Empty Recycle Bin**

  Click **Empty Recycle** Bin to purge all the artifacts in the recycle bin from the repository.

Click an artifact name to open the Artifact Details page for the deleted artifact.

Deleted Artifact Detail Page



Version: 1.0
Provider:
Consumable: No
Service Type: Business service

No tasks defined.

HP Systinet Administrator, 11:38 AM

None set

Service 'CarSalesService' was deleted — New HP Systinet Administrator, Jan 22, 2015 11:38:13 AM

The normal icon for the artifact type is replaced with a wastebasket icon.

The context actions component contains options to **Restore** or **Delete Permanently**.

Chapter 8: Versioning

Services and their associated artifacts sometimes need to be changed or modified to reflect changes in business processes, service dependencies, or other business criteria. Services are composed of metadata, documents, and implementation artifacts. Maintaining proper relationships between all these changing components can be challenging, so Systinet includes extended support for the versioning of services and artifacts. This support includes features such as automatic version resolution during file uploading, and easy version creation based on selected versioning strategies.

Systinet uses both revisions and versions and these different concepts are described in the following topics:

- "Revisions" below
- "Versions" on the next page

There is a close relationship between lifecycle governance and versioning. For details, see "Lifecycle and Versioning" on the next page.

Systinet provides the following functionailty for creating and supporting versions described in the User Guide:

- "How to Create Versions"
- "How to Upload Versioned Data Files"
- "How to Compare Versions"

"Versioning Best Practice" in the *User Guide* describes HP recommendations for your versioning policy.

Revisions

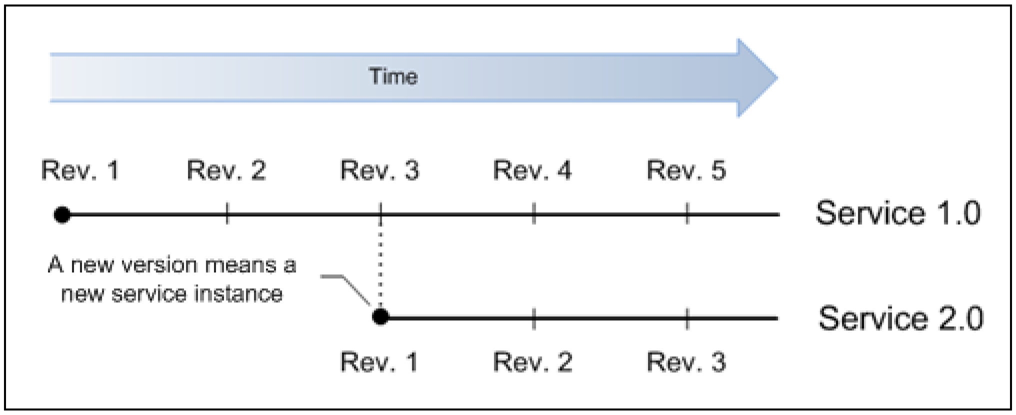
Revisions are used to track changes to individual artifacts and content in the catalog. Systinet stores a complete history of each artifact instance as a revision. Whenever an artifact changes Systinet automatically creates a new revision. There is no versioning schema associated with the tracking of revisions.

You normally work with the latest revisions of artifacts in the UI, but Systinet provides the Artifact Details page History tab, enabling you to view and compare different artifact revisions and see the history of all changes. For details, see "Artifact Details Page History Tab" in the *User Guide*.

Versions

While revisions are used to track the history of changes to services and their underlying artifacts, Versioning is a different concept. All services and artifacts have version numbers assigned to them by Systinet when they are created, or imported to the Catalog. As users create new versions, the versioning process within Systinet implements automatic changes to version numbers for individual artifacts, groups of artifacts, or the entire service, based on the scope of the changes being made.

When a version is created in Systinet then a new branch (newly created artifact instance) is created. The properties of the new branch are automatically set according to the previous version. In other words, as a starting point, the new branch is essentially a clone of the previous version. HP recommends creating a new branch only when it is necessary to have more than one active version (artifact instance) at the same time. The typical example of this would be the need to have multiple implementations of the same service in production.



You can use the Version context actions in artifact detail pages for quick navigation between different versions (branches). For details, see "How to Compare Versions" in the *User Guide*.

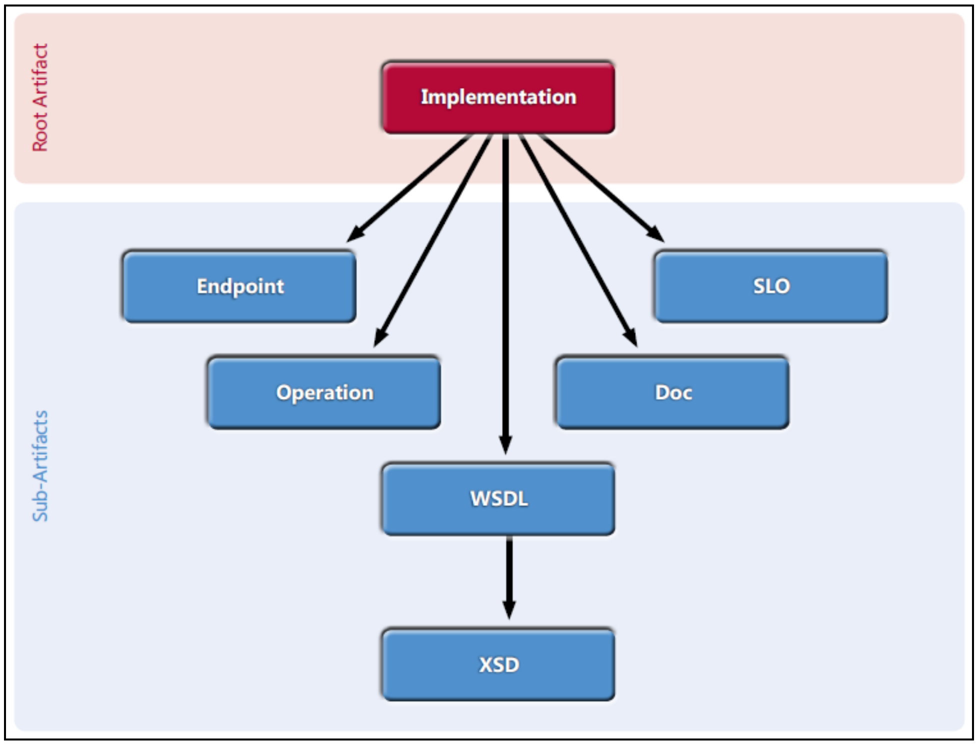
Lifecycle and Versioning

Versioning is a natural consequence of using Lifecycle Governance.

Lifecycle Trees and Versioning

Artifacts that progress through a governance lifecycle together are grouped together in Systinet in a structure called a Lifecycle Tree. Lifecycle trees are defined by the user when setting up a lifecycle and can vary in how they are defined across different lifecycles.

Lifecycle trees are built around the concept of a Root artifact and Sub-artifacts, where there is a single root artifact for any lifecycle tree, with one or more sub-artifacts associated with that root artifact. For example, in the following diagram, the root artifact in the lifecycle tree structure is an Implementation, with sub-artifacts Endpoint, Operation, WSDL, Documentation, and SLO. The WSDL has an additional sub-artifact, XSD.



Because versioning involves potential changes to multiple artifacts in a lifecycle tree, there is a complex relationship between versioning and lifecycle. When a lifecyclee tree (root and sub-artifacts) is assigned to a lifecycle process, versioning is only allowed at the root artifact level; you cannot version sub-artifacts independently of the root artifact, and they always follow root artifact versions. This relationship between root and sub-artifacts in the lifecycle tree is tracked and understood by Systinet, so that the versioning relationship between the root and sub-artifacts is automatically maintained. Users who make changes to the version of a root artifact do not have to manually update the version numbers of the sub-artifacts.

However, not all services and artifacts are placed under governance. In this case, there is no lifecycle tree defined, hence no formal relationship between the different service artifacts. You can version any ungoverned artifact at anytime, but the versioning behavior is different. Unlike the example above, making a change to the implementation version of an ungoverned artifact does not automatically result in an update to the versions of the Endpoints, Operations, WSDLs, etc. The user must apply these changes manually if they want versioning to propagate across other artifacts.

Lifecycle Processes and Versioning

Lifecycle processes drive the versioning process. It is advisable to put artifacts under governance before versioning them, so that the versioning relationships across different artifact types can be maintained automatically by Systinet. This significantly reduces the complexity of managing versioning relationships.

In addition, HP recommends using appropriate lifecycle processes. For example, for the appropriate versioning of service implementations it is crucial to govern services and their implementations independently. For more details, see "Lifecycle" on page 61.

Chapter 9: Contracts

A key service relationship is the provider-consumer relationship that you can create between service consumers and providers.

There are several reasons to capture provider-consumer relationships in Systinet:

- **Change Management**

  By requiring consumers of services to register, the provider ensures that all users of that service are notified whenever some aspect of it changes.

- **Contract Access Control**

  A controlled way of providing consumers access to a service, helping prevent unanticipated or inappropriate usage.

- **Contract Monitoring**

  Monitors how successful a particular service is, based on its level of reuse and its use across organizational units.

- **Contract Auditing**

  Providers need to know who is using specific services and who granted them access. This could be for many reasons, such as security, accounting, capacity planning, or change management.

Contract management controls the contract of services by defining the following:

- **Consumer**

  The consumer is the person or organizational unit that uses a service.

- **Provider**

  The provider is the user or organizational unit responsible for contracts associated with the artifact that the consumer wants to use.

- **Contract Request**

  When a consumer finds an artifact that they want to use, they create a contract request which is submitted to the provider.

- **Contract**

  When a request is approved, a contract is formed between the consumer and provider artifacts.

- **Service Level Objectives**

Each provider artifact can be associated with a set of offerings that the provider sets out as performance levels for the service. When a consumer makes a request they can select an SLO as part of the terms of the contract.

There are two SLO artifact types:

- Service Level Offering artifact
- Service Level Objectives artifact

Service Level Offering artifact:

- It will be created by provider
- It will be added into provider artifact

Service Level Objectives artifact:

- It is a copy of **Service Level Offering** artifact.
- It will be created when consumer create a contract and add a SLO artifact for contract.

Contract Management consists of the following processes described in the *User Guide*:

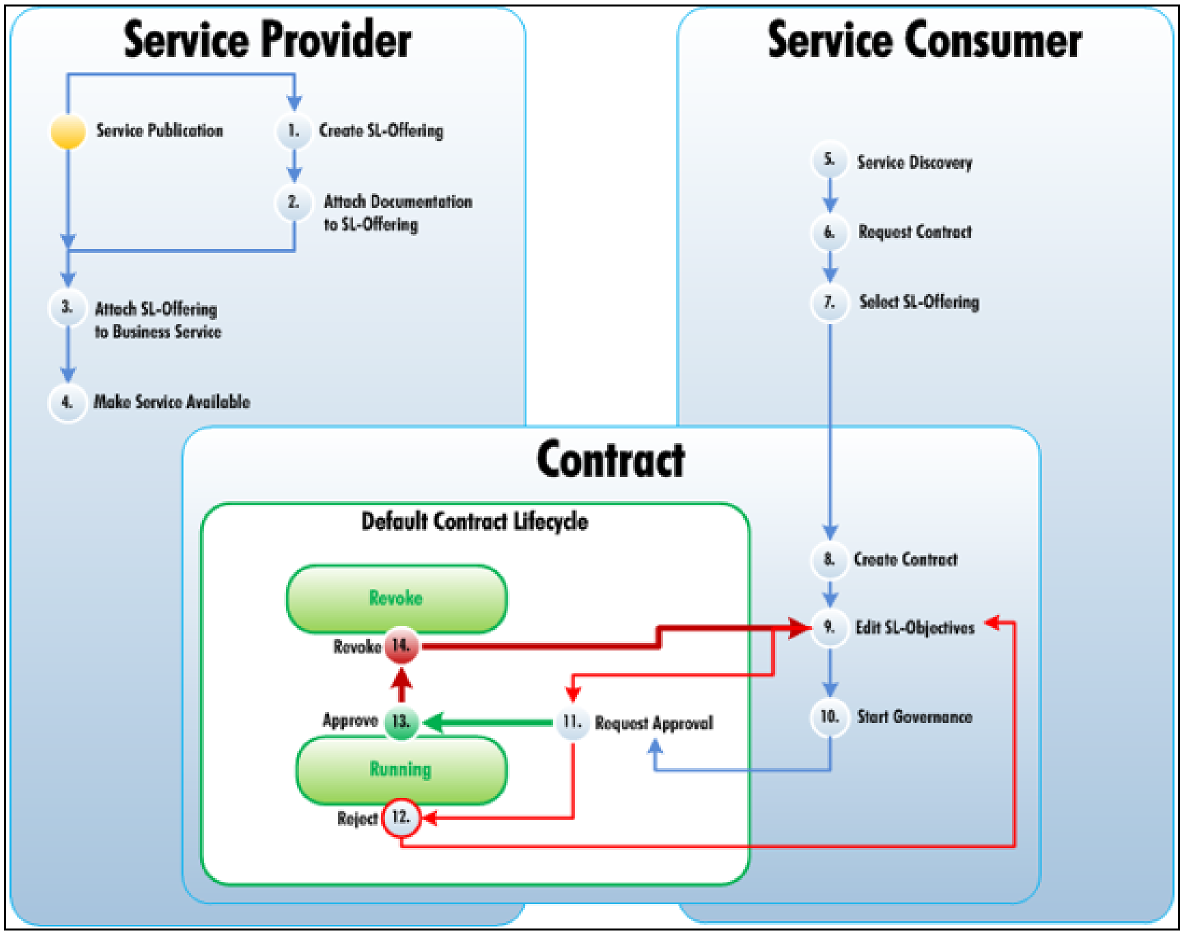
Contract Management fits onto the larger service provision use case as described in "Contract Workflow" below.

Systinet provides a default set of artifact types that can be consumer or provider artifacts. The administrator can customize these settings. For details, see "Default Provider / Consumer Artifact Types" in the *User Guide* and "Configuration Management" in the *Administration Guide*.

Contract Workflow

For a provided service to be consumed, there are typically, several steps required by the provider and a discovery process for the consumer. Having found a service that meets their requirements the consumer and provider can establish a contract. The “Contract Workflow” diagram displays the typical tasks that should be performed, and an overview of each step is given below.

Note: The workflow uses a service as the example. Systinet enables other artifact types to be providers as well. For details, see "Default Provider / Consumer Artifact Types" in the *User Guide*.



1. After you create a service, it is normal practice to offer a number of service-level objectives describing the terms of use for the service. You can create an SLO independently, or as part of a service. For details, see "How to Create Artifacts" and "Create SLO Page" in the *User Guide*.

2. You can also attach documentation to the SLO. For details, see "How to Attach Documentation" in the *User Guide*.

3. Attach the SLO to the Service. For details, see "How to Manage Relationships" in the *User Guide*.

4. Make the Service available by marking it as consumable. For details, see "How to Edit Artifact Properties" in the *User Guide*.

   Note: This can be automated as part of a lifecycle process. For details, see "Automatic Actions" on page 65.

5. The consumer browses Systinet to see which services are available and whether they meet their requirements. For details, see "Search and Browse" on page 15.

6. After the consumer finds the service, they can request it. For details, see "How to Request Contract" in the *User Guide*.

7. When a consumer requests consumption of a service, you must either approve or reject the request. For details, see "How to Process Contract Requests" in the *User Guide*.

Contract Management as Lifecycle Process

Previous versions of Systinet used special artifacts SDM model to model contract management. Systinet 10.01 obsoletes part of contract management SDM and implements contract management functionality on top of more powerful and flexible lifecycle process to govern directly contract artifacts. Administrators can create new lifecycle process to govern contracts and customize stages, approvals, actions according to their corporate standards.

Chapter 10: Runtime Gateway Interoperability Framework (RGIF)

This part contains the following topics:

Concepts

- "Runtime Contracts" on page 59
- "Manual Proxy Creation and Contract Deployment"
- "Lifecycle Based Contract Development" on page 56
- "Composite Application Development Lifecycle" on page 56

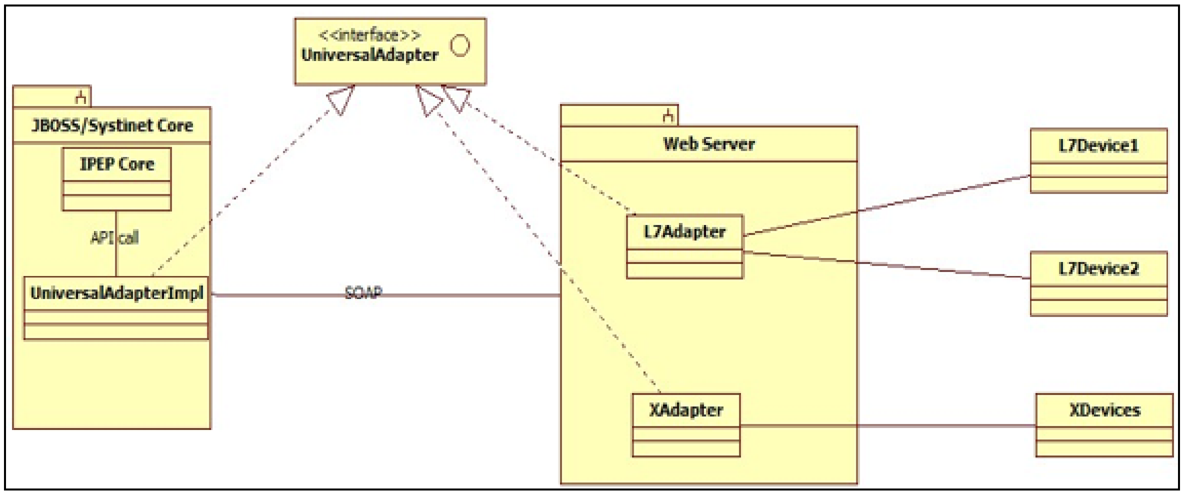
Runtime Gateway Interoperability Framework:

- Remote implementation (adapter)
- RGIF discovery
- REST service deployment

Remote implementation (adapter)

RGIF implementation is decoupled from Systinet 10.01 codes. Integration between Systinet and PEP (Policy Enforcement Point) is now communicated through a RGIF interface.

Systinet 10.01 contains RGIF implementations for Layer7 (all versions) and Datapower (v6) OOTB.



RGIF discovery

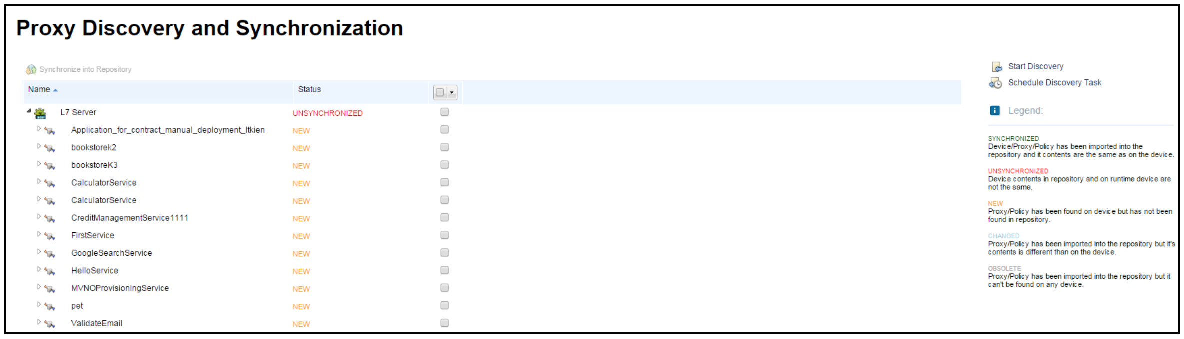
Systinet 10.01 RGIF provides ability to discover REST and Web services from PEPs.

From runtime devices, Web services are discovered and synchronized to Systinet as SOA artifacts, while REST services are synchronized as API artifacts. Discovered artifacts could then be governed within Systinet or deployed to different runtime devices or environments.

Policies from runtime devices are also discovered, decomposed and synchronized into Systinet as RGIF artifacts (proxy, universal policy and policy parameter artifacts).

Discovery UI

Discovery UI is introduced for manual discovery by administrator (or domain administrator).



Proxy Discovery and Synchronization

Synchronize into Repository

Name	Status	
L7 Server	UNSYNCHRONIZED	☐
Application_for_contract_manual_deployment_ltkien	NEW	☐
bookstorek2	NEW	☐
bookstoreK3	NEW	☐
CalculatorService	NEW	☐
CalculatorService	NEW	☐
CreditManagementService1111	NEW	☐
FirstService	NEW	☐
GoogleSearchService	NEW	☐
HelloService	NEW	☐
MVNOProvisioningService	NEW	☐
pet	NEW	☐
ValidateEmail	NEW	☐

Start Discovery
Schedule Discovery Task

Legend:

SYNCHRONIZED
Device/Proxy/Policy has been imported into the repository and it contents are the same as on the device.

UNSYNCHRONIZED
Device contents in repository and on runtime device are not the same.

NEW
Proxy/Policy has been found on device but has not been found in repository.

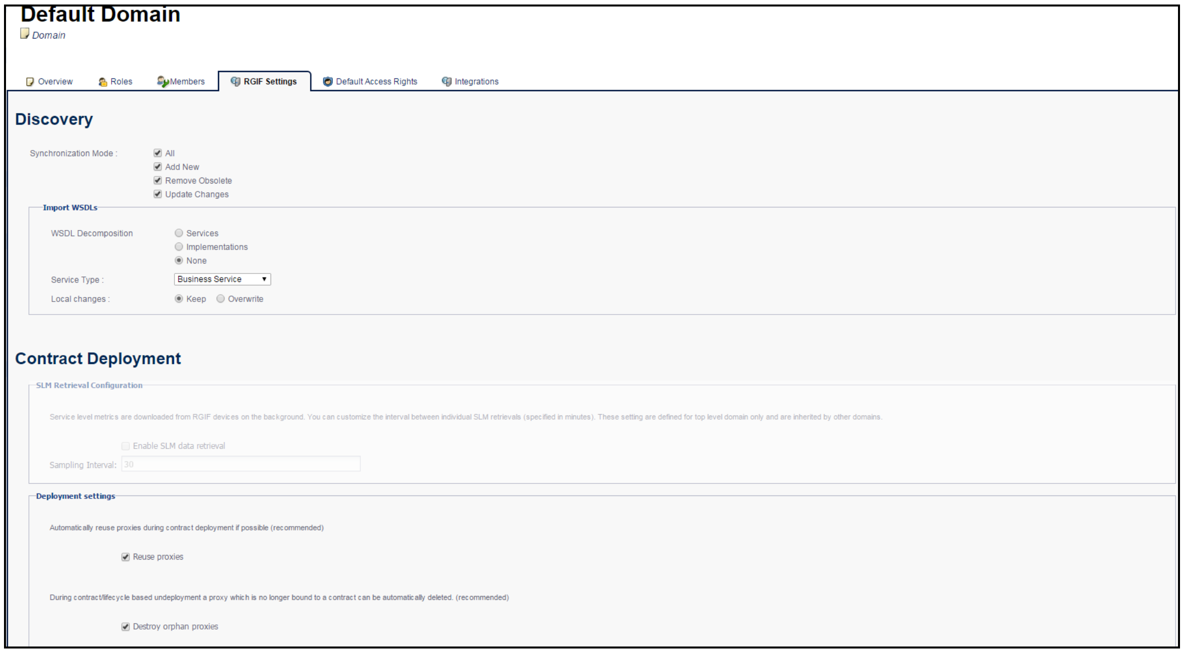
CHANGED
Proxy/Policy has been imported into the repository but it's contents is different than on the device.

OBSOLETE
Proxy/Policy has been imported into the repository but it can't be found on any device.

Discovery Task

Discovery task is introduced for automatic discovery.

Administrator first specifies whether to decompose WSDL of runtime services and synchronization mode by using **platform.rgifdiscovery.wsdls.decomposition** and **platform.rgifdiscovery.synchronization.mode** system properties or by the configuration.



Default Domain
Domain

Overview | Roles | Members | **RGIF Settings** | Default Access Rights | Integrations

Discovery

Synchronization Mode : [✓ All] [✓ Add New] [✓ Remove Obsolete] [✓ Update Changes]

Import WSDLs

WSDL Decomposition: Services / Implementations / [✓ None]

Service Type : Business Service

Local changes : [✓ Keep] Overwrite

Contract Deployment

SLM Retrieval Configuration

Service level metrics are downloaded from RGIF devices on the background. You can customize the interval between individual SLM retrievals (specified in minutes). These setting are defined for top level domain only and are inherited by other domains.

☐ Enable SLM data retrieval

Sampling Interval: 30

Deployment settings

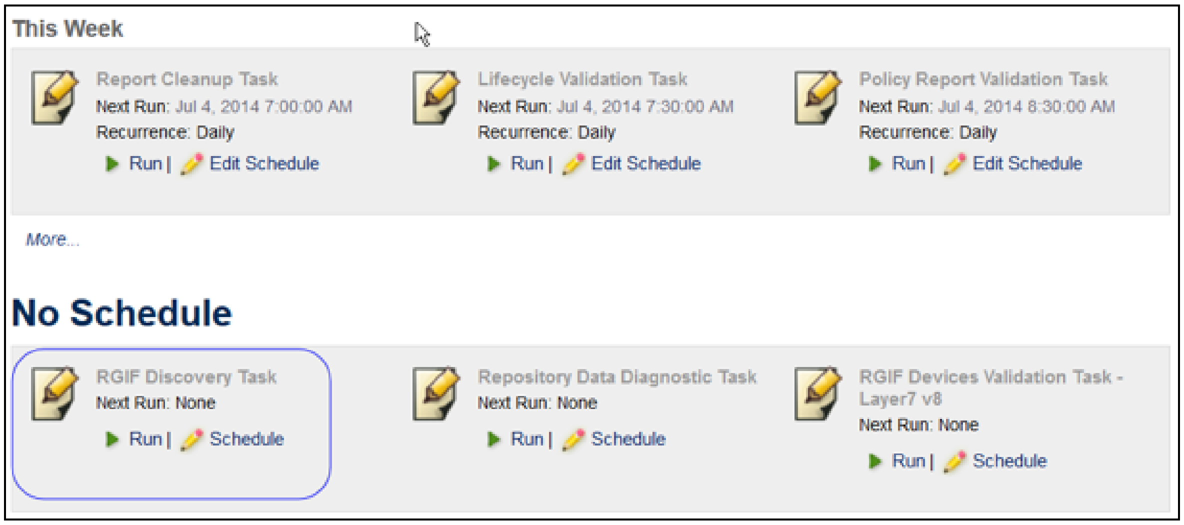
Automatically reuse proxies during contract deployment if possible (recommended)

[✓ Reuse proxies]

During contract/lifecycle based undeployment a proxy which is no longer bound to a contract can be automatically deleted. (recommended)

[✓ Destroy orphan proxies]

Then set schedule for RGIF Discovery Task to run at a regular interval.


This Week

Report Cleanup Task
Next Run: Jul 4, 2014 7:00:00 AM
Recurrence: Daily
Run | Edit Schedule

Lifecycle Validation Task
Next Run: Jul 4, 2014 7:30:00 AM
Recurrence: Daily
Run | Edit Schedule

Policy Report Validation Task
Next Run: Jul 4, 2014 8:30:00 AM
Recurrence: Daily
Run | Edit Schedule

More...

No Schedule

RGIF Discovery Task
Next Run: None
Run | Schedule

Repository Data Diagnostic Task
Next Run: None
Run | Schedule

RGIF Devices Validation Task - Layer7 v8
Next Run: None
Run | Schedule

RGIF deployment

REST service deployment - Repository users could deploy REST services from Systinet 10.01 to PEP as they used to doing with Web services in Systinet 4.10.

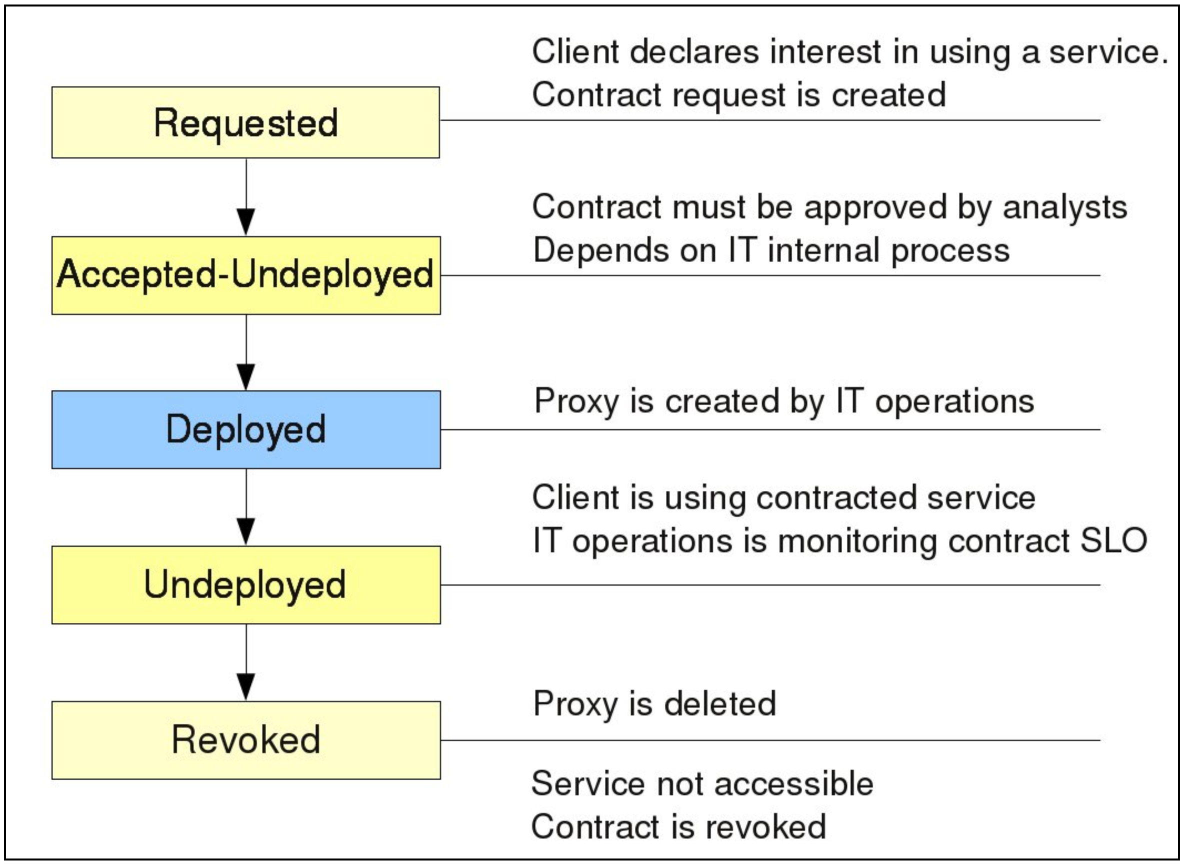
Composite Application Development Lifecycle

Composite applications are those that are formed from multiple services, are captured within the Systinet repository and have contracts created between them. Composite applications may have their own dedicated lifecycle process and if this is correctly defined it can be used to move composite application contracts and proxies created in these lifecycle processes from one environment to another – Systinet automatically creates these proxies in the new environment.

See "Lifecycle" on page 61 for more information about lifecycle processes.

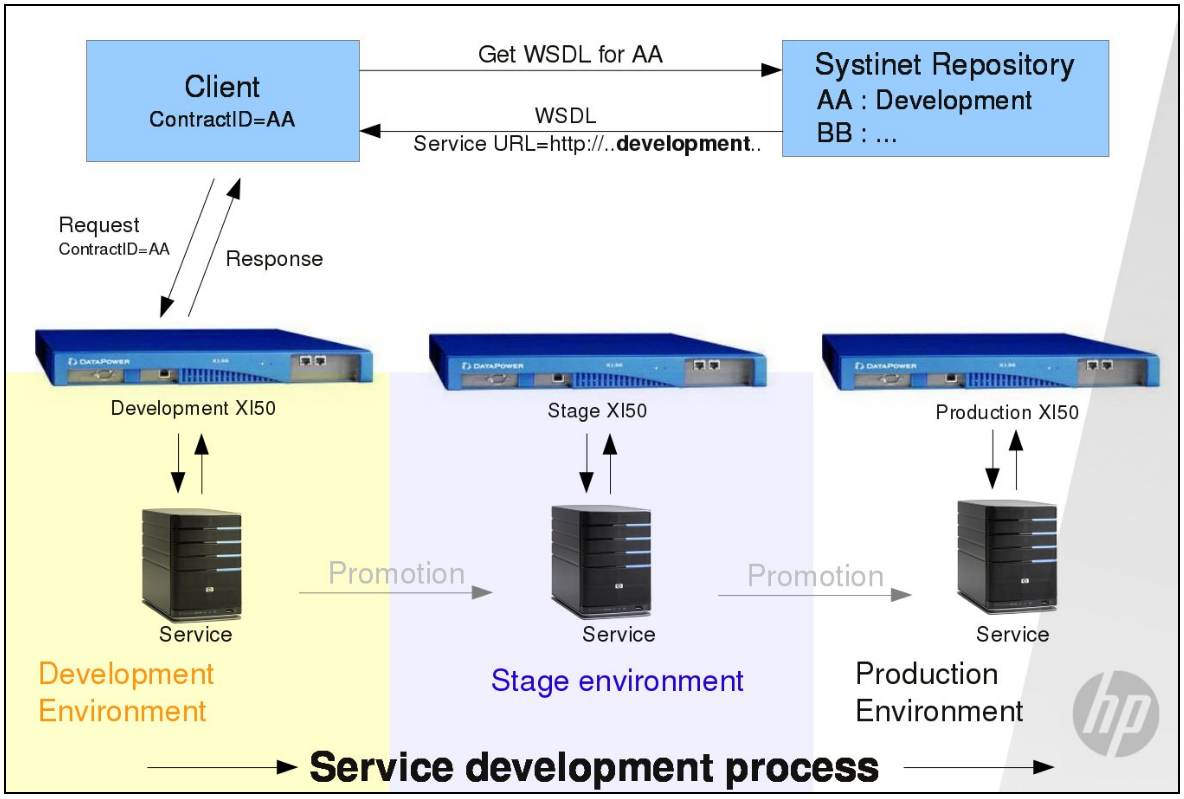
Lifecycle Based Contract Development

Proxy objects can be also created during an contract artifact lifecycle by assigning Create/Remove PEP Proxy automatic action assigned to a lifecycle stage. The most simple lifecycle process looks like the following:


Requested
Accepted-Undeployed
Deployed
Undeployed
Revoked
Client declares interest in using a service.
Contract request is created
Contract must be approved by analysts
Depends on IT internal process
Proxy is created by IT operations
Client is using contracted service
IT operations is monitoring contract SLO
Proxy is deleted
Service not accessible
Contract is revoked

Note that the “Requested” state is represented by a contract request artifact (see "Contracts" on page 50 for more information) and the contract lifecycle starts in the “Accepted-Undeployed” state. It is useful to configure the lifecycle process to be automatically assigned to contracts.

The above lifecycle might be too simple to be used in a typical IT environment where there are different (separated) environments/landscapes for service development/testing/production. Systinet supports this scenario as well and all that is required is to associate RGIF devices with environments and define the contract lifecycle in a similar fashion as likely done for the service lifecycle process – there will be stages matching the environments defined in the lifecycle.

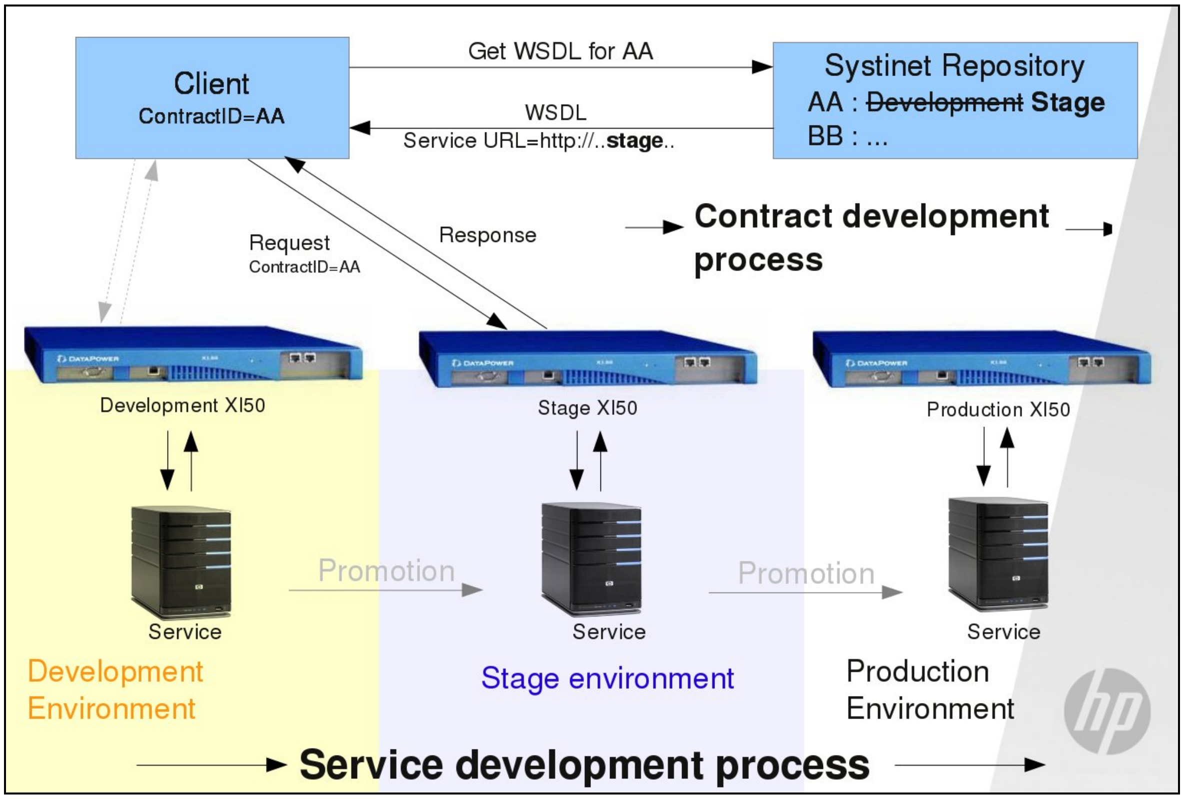


The contract lifecycle has stages matching the service lifecycle and it means the following – when a contract is in the development environment lifecycle stage it is accessing managed endpoints created for the service hosted in the development environment. In the image above the contract AA is in the Development environment lifecycle stage .

The process is as follows:

1. The client creates a WSDL request on the Systinet Repository passing his contract ID.
2. Repository returns WSDL with the service url pointing to the RGIF device assigned to the Development environment
3. Based on the information in the WSDL the client sends a message to the proper proxy object hosted on the RGIF device and the response is returned

When the contract lifecycle is promoted to the next lifecycle state the client acts in the very same way but will receive a different URL within the WSDL.



The important and valuable aspect about this is that contracts are moved between environments just by changing the information in the repository; client reconfiguration is not required. This becomes useful when the client is deployed multiple times on different computers as in the example of a desktop application being used by multiple users.

Also note that you can have a single contract deployed across multiple environments; in this case the client must pass except the contract ID also the environment which is intended to be accessed.

See "Lifecycle" on page 61 and "Automatic Actions" on page 65 for more information about lifecycle processes.

Manual Proxy Creation and Contract Deployment

Proxy objects can be created on demand directly from the repository user interface using a dedicated UI wizard. This means of proxy creation/contract deployment is meant to be used primarily by administrators to fix exceptional states when an automated proxy creation within a lifecycle fails. In some situations where full lifecycle process is not required, it might be used by regular users to create proxies. This feature must be explicitly enabled in the general contract deployment settings.

Runtime Contracts

Because proxy objects are created on behalf of a contract, you can apply contract enforcement. This means that only messages sent by negotiated service consumers that are configured in the repository are allowed to send messages to the application endpoints. This is the default behavior of the proxy objects created by Systinet.

RGIF devices need to perform contract identification to ensure contract enforcement. In proxy objects, it is encoded which contracts are allowed to send messages to the proxy object. The identification itself is performed based on message properties such as an HTTP header holding the contract ID sent within each message. There are some other means to perform contract identification; HTTP basic credentials, XPATH expression evaluated over the message and others.

Proxy objects created by Systinet also allow per contract service monitoring, which means that statistics such as the number of messages sent per day by an individual contract may be monitored.

See "Contracts" on page 50 for more information about contracts.

Chapter 11: Lifecycle

Artifacts have several lifecycle stages ranging from being merely a candidate through development, implementation, and eventual deprecation and reuse. Each stage has own specific features and each organization has different detail requirements for these different stages. The stages can be divided into development and runtime stages and before a service can be allowed to move from one stage to another, all necessary policy requirements and approvals must be in place. See "Lifecycle Tree" on the next page for details.

In Systinet lifecycle processes are defined and given policy, task and approval requirements by an administrator. See "Lifecycle Process Management" in the *Administration Guide* for details.

These processes are then either automatically or manually applied to artifacts. See "Automatic Assignment" on page 69 and "Lifecycle Best Practice" in the *Administration Guide* for details.

Manual lifecycle tasks can be assigned to different users and can have policies that must be validated before task is completed. Policies as well as permissions can be associated with the lifecycle process stage. See "Lifecycle Tasks" on page 64, "Permissions" on page 66, and "Policy Management" in the *Concepts Guide* for more details.

When all the requirements and tasks are complete, the artifact owner makes a request to move the process to the next stage. If the administrator has assigned approvers, they are notified and are required to vote on approval. Depending on transition type, the governed artifact moves to the next stage and the lifecycle automatic actions defined for these stages are triggered. See "Approvers" on page 66, "Stages and Transitions" on the next page, and "Automatic Actions" on page 65 for more details.

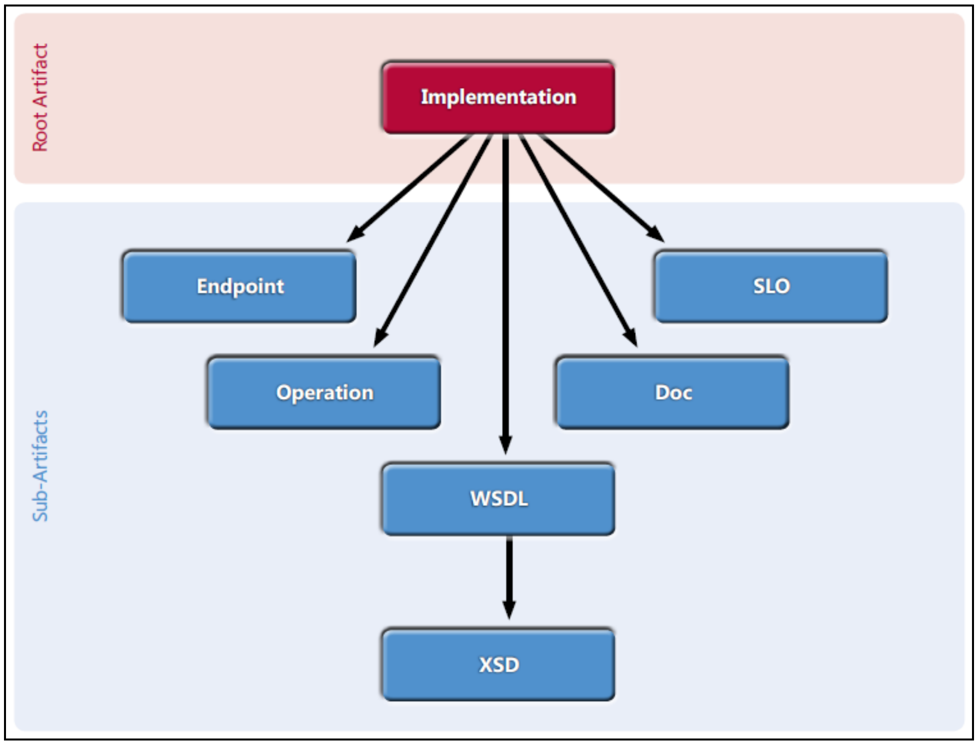
For procedural topics please see the *User Guide*:

- "How to Start Governance"
- "How to Review/Close Lifecycle Tasks"
- "How to Review/Validate Policies"
- "How to Review Compliance Status"
- "How to Request Approval"
- "How to Approve/Reject Approval Requests"
- "How to View Lifecycle History"
- "How to End Governance"

Lifecycle Tree

Artifacts that progress through the governance lifecycle together are grouped in Systinet by a structure called a Lifecycle Tree. Lifecycle trees are defined by the user when setting up a lifecycle process and can vary in how they are defined across different lifecycle processes.

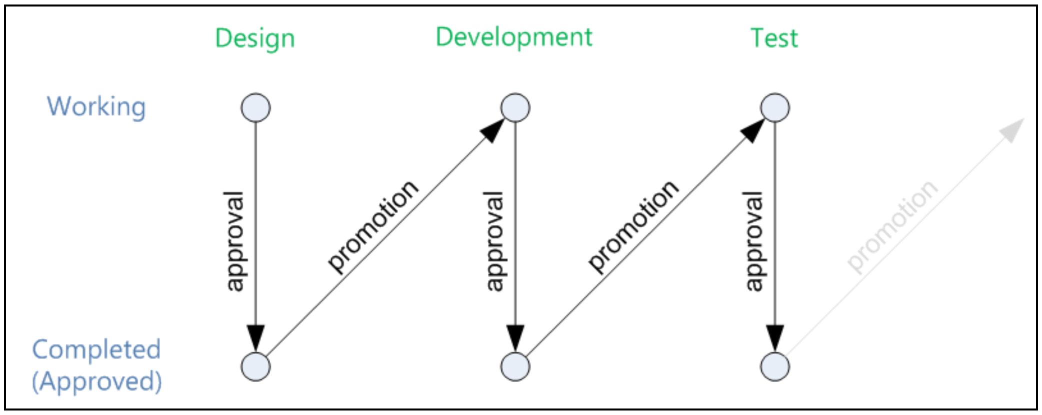
Lifecycle trees are built around the concept of a Root artifact and Sub-artifacts, where there is a single root artifact for any lifecycle tree, with one or more sub-artifacts associated with that root artifact. For example, the root artifact in the lifecycle tree structure below would be Implementation, whereas sub-artifacts are Endpoints, Operations, WSDLs, XSDs, Documentation, and SLOs.



Stages and Transitions

A lifecycle consists of a set of stages and transitions between them.

The following image shows part of the lifecycle process:



The current stage of a service in a lifecycle can be described and queried in the following ways:

- **Working Stage**

  The working stage represents the stage currently in progress. These are stages in which there may be tasks to be completed and/or policies to be validated before stage can approved for advancement.

  A working stage can have different status:

  - In Progress - Work is being performed on the stage.
  - Voting - Stage is waiting for assigned approvers to vote.
  - Rejected - Stage has been voted against being approved by an assigned approver.
  - Failed - An assigned and required policy constraint has not been met.

  See "Approvers" on page 66 and "Policy Management" in the *Concepts Guide* for more information.

- **Approved Stage**

  The approved stage represents the last completed stage in which all tasks and policies have been completed and has been approved.

Moving between stages involves the following lifecycle transitions:

- **Approval**

  The approval process is the confirmation that all the requirements for the current Working Stage are complete and results in the Approved Stage.

- **Promotion**

  Promotion initiates the next Working Stage. This requires the current Working Stage to be Approved and can be automated as part of the process definition.

There are different types of transitions that are described in "How to Define Transitions" in the *Administration Guide*. There may also be a set of automatic actions associated with a lifecycle events as described in "Automatic Actions" on the next page.

Lifecycle Tasks

A task specifies the execution of an action against a lifecycle stage. These tasks are tracked and they must be marked as completed before starting the approval process.

Tasks may be assigned to users, groups or roles.

Lifecycle process is applicable to contract artifact, tasks can be assigned to contact. Task contact assignee includes:

- **Maintainer(s) of the provider** task contact assignee, this assignee is a list of users, roles, groups which have **WRTIE** permission on provider artifact. This list will be automatically filled when requesting approval and list's number required is one.
- **Contact role** task contact assignee includes Provider of the provider(s), Developer of the provider (s)", and so on and this assignee is an user who is a contact of provider of consumer artifact.

Examples of common tasks:

- Requesting that documentation is attached to an artifact such as business requirements
- Requests for deployment
- Requests for builds
- Requests for testing

Technical policies applicable to the root artifact may be associated with tasks Systinet automatically validates these policies when of a task is marked as complete. If the policy fails validation, the user is notified in the application and the task is not considered complete.

See "Lifecycle Policies" below for more information about policies.

Lifecycle Policies

Lifecycle stages in a lifecycle process can have a set of policies associated with them that serve as validation checks for artifacts at that lifecycle stage. Policies can be optional or required and can also validate that tasks associated with the stage are complete. These policies are automatically validated when a user requests stage approval or can be manually validated by users. The collective status of these policies gives a Compliance Status for an artifact which measures the current percentage of required policies which pass validation. The administrator assigns policies to lifecycle stages during the creation of lifecycle processes. For details about setting up and performing lifecycle validation, see "How to Define Policies" in the *Administration Guide* and "How to Review/Validate Policies" in the *User Guide*.

Systinet includes a default Lifecycle Validation Task, scheduled to run once a day, which automatically validates all artifacts in governance against the policies that apply to their current lifecycle stage. For details about managing this task, see "Administration Task Management" in the *Administration Guide*.

Automatic Actions

At each lifecycle stage, you can define a set of Automatic Actions to modify an artifact when certain lifecycle events occur.

The available automatic actions are:

- **Execute Script**

  Invoke a custom javascript code previously defined using administration, customization, and manage scripts. In the Advanced tab, you can define a javascript fragment, which is executed prior to running the referenced script artifact.

- **Notify**

  Send automated mail notifications to sets of users as a result of a lifecycle change. For example, notify consumers that a new version of a service is approved in the Production stage and is available for consumption.

- **Deploy/Undeploy Contract**

  Deploy or undeploy a contract. Deploy action also updates an existing proxy if available. For example, you can deploy a contract to a Layer 7 server when a contract in the Production stage is approved.

- **Update Proxies**

  Rebuilds PEP proxies for contracts of the approved service or contract provider. For example, the proxies of contracts associated with the governed artifact will be recreated when a contract in the Production stage is approved.

- **Enable Consumption or Disable Consumption**

  Allow or disallow the creation of contracts for an artifact as a result of a lifecycle change. For example, set the Consumable flag when a Service in the Production stage is approved. This allows the artifact to be used in a contract and make it visible for the partner role.

- **Export to Registries or Delete from Registries**

  Either exports or deletes an artifact from registries. This would be used if registries contained artifacts for specific purposes i.e.: development, production, etc.. Once artifacts move from one stage to another, it may be desired that they be removed or added to different registries.

- **Parent Request Approval**

  Creates an approval request for a parent artifact. Allows the tying together of processes so that a parent implementation can have an automatic approval requested when the dependencies have been approved.

- **Remove Comments**

  Removes all comments from the artifact when moved to next stage. This may be used when there is some debate about functionality within the commentary, but once resolved, does not need to be retained when moving to the next stage.

Approvers

At each lifecycle stage, you can define a set of approvers whose votes are required to approve the current stage. These votes can be assigned to users, groups or roles. If the specified number of votes is not met by a group or a role, or there is a negative vote, there will be no approval and the stage will not advance.

Lifecycle process is applicable to contract artifact, votes can be assigned to contact. Contact voter includes:

- Maintainer(s) of the provider contact voter, this voter is a list of users, roles, groups which have **WRTIE** permission on provider artifact. This list will be automatically filled when requesting approval and list’s number required is one.

- **Contact role** contact voter includes Provider of the provider(s), Developer of the provider(s) and this voter is an user who is a contact of provider of consumer artifact.

It is also possible to allow passive approval which is an automatic approval after a specified time period. This may be used if approval is not mandatory but still needs to be visible in the application to show responsibility.

See "How to Define Approvers" in the *Administration Guide* for more information on the procedure for setting of passive approval.

Permissions

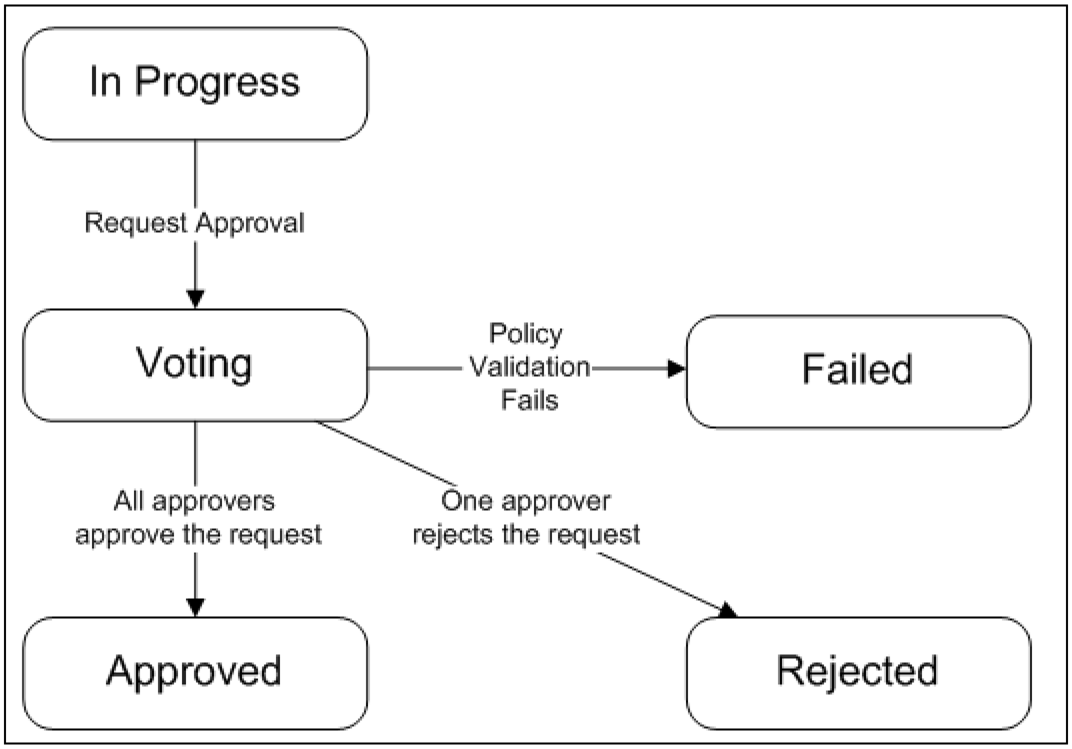
Each stage in a lifecycle process has a user(s) assigned different Permissions for that stage allowing them to manipulate the stage based on predefined security settings. These settings are initially empty and are defined by the administrator to include read/write access, ownership, assignment as an approver for advancing the stage and the option to share the artifact globally during a specified stage. These settings are only for the assigned stage and are not carried forward to the next stage.

Note: In only the first stage, domain users are granted read permission by default.

See "How to Define Permissions" in the *Administration Guide* for details about setting stage permissions.

Lifecycle Approval

The following image depicts the lifecycle of each lifecycle stage. When an artifact enters a stage then the stage has the status *In Progress*. After all tasks have been completed , a user requests approval and the stage status is changed from *In Progress* to *Voting*. At the beginning of *Voting*, the system checks whether all required policies are complaint. If not, the status is changed from *Voting* to *Failed*. The requestor is informed about the failed policy validation and he must repair the artifact and then request approval again. Once all required policies are compliant, approvers are notified that their vote is required. When all approvers approve the request, the stage status is changed from *Voting* to *Approved*. If any approver rejects the request, the request is rejected (regardless of the fact that some approvers may approve it) and the stage status is change from *Voting* to *Rejected*. The requestor is notified about the rejection and he must repair the artifact(s) and request approval again.



The following list describes the main activities for each stage status.

- **In Progress**

  User primarily wants to have the artifact approved in the current stage and so he wants to request approval. Before he can request approval, all tasks must be completed and required policies should be compliant. If any required policy is non-compliant then the user can ask for approval, but the system checks the compliance of policies at the beginning of the voting and status is automatically changed to *Failed*.

  Tasks can be assigned to specific users (e.g. via roles), but task assignment does not block task completion. This means that even users who do not have the task assigned can complete it. The system tracks who completes the task and when. If a policy is assigned to the task, the task cannot be marked as finished if policy validation fails.

  In addition to requesting approval, a user can move artifact to next stage, but only stages connected via the transition *Manual Anytime* type can be used.

- **Voting**

  The artifact is locked for editing during voting.

  Approvers are notified (via mail and events) about a new request waiting for their vote. They can either approve or reject the request. Before an approver casts his vote he can view any changes that happened in the stage; i.e. he can compare the last approved revision with the current one via differences. Approvers approve or reject not only the root artifact but also all sub-artifacts from the lifecycle tree.

  Users and approvers can view the voting status. They can find out which approvers approved the request, read approvers' comments, and when the approval was given. They can also view which approvers did not vote.

  At the beginning of the voting process, policies are checked and if any required policy is non-compliant then approvers are not notified about the request. The request is automatically canceled, its status changed to *Failed* and the requestor is notified via mail and events.

- **Approved**

  In a perfect scenario, all approvers approve the request and so the request is approved. The stage status is changed from *Voting* to *Approved* and the requestor is notified about the approval via mail and events.

When an artifact(s) is approved and if there is a transition of *Automatically, When Candidate is Approved* type (e.g. in initial lifecycle stages) then the artifact(s) is automatically moved to the next stage and so the *Approved* status of one lifecycle stage is triggering the automatic change to the *In Progress* status of the next lifecycle stage.

If approved artifact from the lifecycle tree is changed then the status is changed from *Approved* to *In Progress*.

- **Rejected**

  When any approver rejects the request, the request is rejected and the stage status is set to *Rejected*. The requestor is notified about the rejection via mail and events, and must repair the artifact and request approval again.

- **Failed**

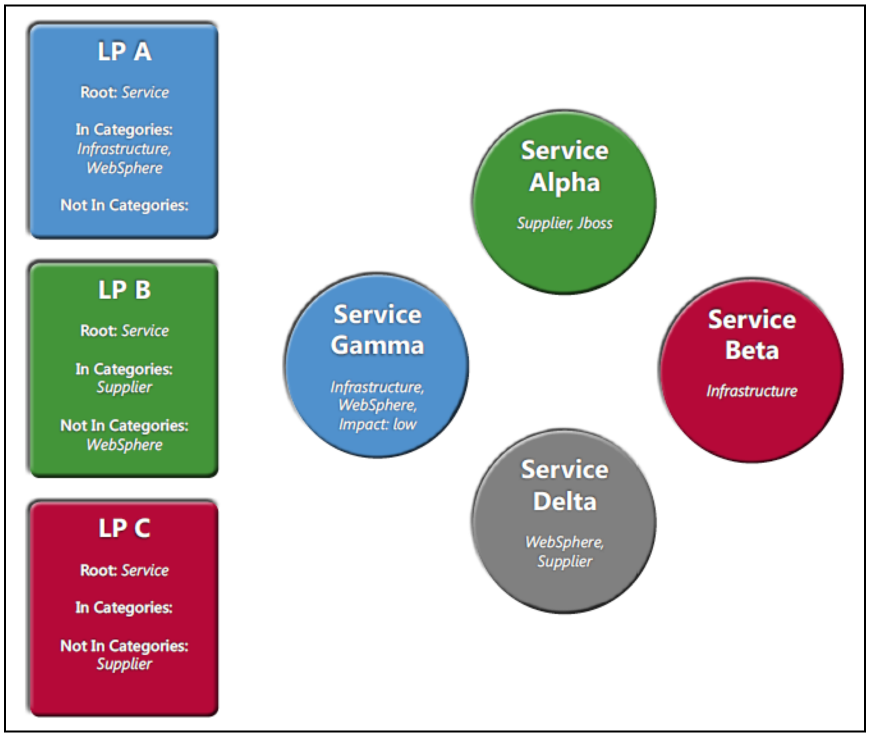
  As mentioned above, when any required policy is non-compliant, the status is set to *Failed*. The user is notified about this by events and must repair the artifact and ask for approval again.

Automatic Assignment

Automatic Assignment applies Lifecycle processes to ungoverned new or recycled artifacts and functions in the following way:

1. Systinet checks that there are lifecycle processes for the artifact type which are marked as **Automatically Assigned.**
2. Systinet checks that the artifact matches the following conditions:
   - The artifact is a root artifact type for the lifecycle process.
   - **In Categories**: The artifact must contain these categories to be assigned to the lifecycle process.
   - **Not In Categories**: The artifact must not contain these categories to be assigned to the lifecycle process.
3. The artifact is assigned to a lifecycle process as follows:
   - If there is no matching lifecycle process, the artifact is not assigned to any lifecycle process.
   - If there is one matching lifecycle process, the artifact is assigned to it.
   - If there are two or more matching lifecycle processes, the artifact is assigned to the most

appropriate one based firstly on the number of matching **In Categories**, and then the number of matching **Not in Categories**.



- Service Alpha is assigned to Lifecycle Process B because of the *Supplier* category match.
- Service Beta is assigned to Lifecycle Process C. It cannot be assigned to Lifecycle Process A because it is not categorized as *Websphere*.
- Service Gamma is assigned to Lifecycle Process A because of the *Infrastructure* and *Websphere* category matches.
- Service Delta cannot be assigned to a lifecycle process so it remains unassigned. Lifecycle Process A specifies that an artifact must be **In Category** *Infrastructure* and Lifecycle Processes B and C both specify **Not In Categories** used by the service.

Tip: Ideally, all artifacts are automatically assigned. See "Lifecycle Best Practice" in the *Administration Guide* for more information.

Chapter 12: Product Integration

Systinet integrates with other products to serve as a central point of governance and to view all the information about your content.

Product Integration covers the following main uses cases:

- Discover and import the content of the integrated products so that you can govern and provide that content within Systinet.
- Publish and share the content of the Catalog with integrated products so that you can apply the features of those products.
- Access the information provided by the integrated products within Systinet.

The administrator is responsible for managing product integration within Systinet. For details, see "Product Integration Management" in the *Administration Guide*.

The exact details for each product vary but server management is handled in a generic way. The administrator creates a server artifact using a URL and a set of default credentials. Depending on the product and the configuration of Systinet, other users can use these default credentials to access the server or store their own in their credentials store. For details, see "How to Manage Integrated Products" in the *Administration Guide* and "How to Manage Your Credentials" in the *User Guide*.

Systinet provides integration and support for the following products:

- **UDDI Registry**

  Use UDDI Registries as central storage for your service infrastructure or for specific environments such as development and production registries. Integration with UDDI Registries enables integration with other products (for example, SAP) through the registry.

  You can import, export, and synchronize your service artifacts with entities in the registry. For details, see "UDDI Registry Integration" on page 75.

- **HP Business Service Management**

  Discover services stored in *HP Business Service Management* (BSM) and *Universal Configuration Management Database* (UCMDB) and enter them into governance. These discovered services are then monitored for changes and you can synchronize any changes from UCMDB into Systinet.

  Use BSM to monitor the performance of your services and view information generated by BSM on shared services in service detail pages. For details, see "BSM / UCMDB Integration" on page 76.

  Note: HP Business Availability Center (BAC) 8.02 is no longer supported.

- **HP Application Lifecycle Manager**

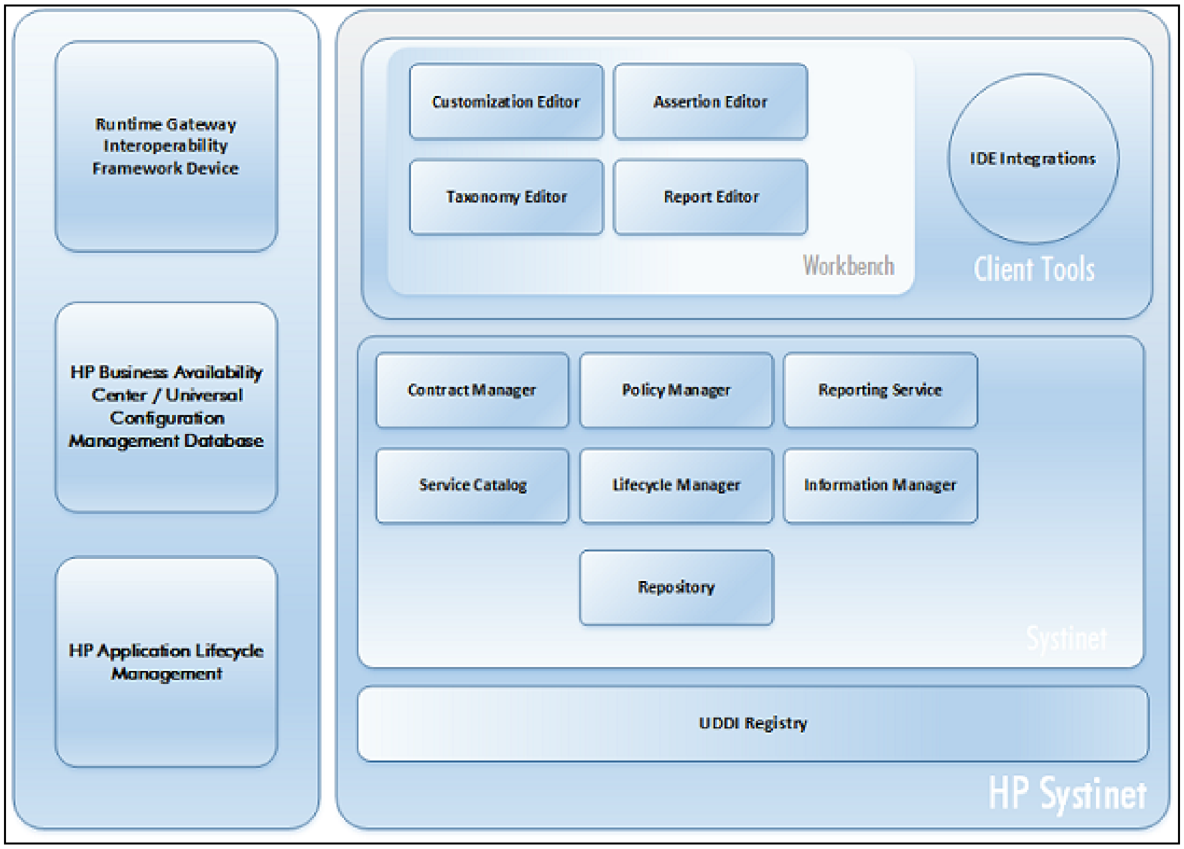
Use *Application Lifecycle Manager* to track your service testing and view information generated by ALM on shared services in Systinet. For details, see "ALM Integration" on page 77.

Systinet also integrates with *LDAP* for user management and *SiteMinder* for user authentication. The management of these integrations is administered during installation or with the Setup Tool. For details, see "Setting Up LDAP Integration" and "Setting Up SiteMinder Integration" in the *Administration Guide*.

HP Software provides plugins for IDEs that enable the use of Systinet features within development environments. In particular, Systinet Plugin for Eclipse can apply to 3rd-party Eclipse-based tools (such as *SAP Netweaver Developer Studio* and *Tibco Business Studio*) as a further integration point. For more details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

Systinet Product Family

Systinet is part of a larger family of SOA products designed to cover the entire range of SOA requirements.



Standard Components

Systinet is designed to enable application governance to be established in a production environment with the following standard components:

- **Catalog**

  Provides an entry point for the provision and management of services.

- **Information Manager**

  Provides a suite of governance tools and low level access to the repository.

- **Reporting Service**

  Monitors and reports on your data.

- **Repository**

  Organizes and manages your data.

- **Lifecycle Manager**

  Enables you to control the service lifecycle and establish governance procedures for each lifecycle stage.

Optional Components

In addition, your Systinet license can include the following components:

- **Policy Manager**

  Enables your environment to conform to your business policy.

- **Contract Manager**

  Enables the management of provider-consumer relationships.

Tools

HP Software provide a set of tools for use with Systinet:

- **Workbench**

  A suite of editor tools distributed as an Eclipse development platform or as a plugin for Eclipse.

  Workbench includes the following components:

  - **Customization Editor**

    Enables you to customize the architecture model and in Systinet. For details, see the *Customization Editor Guide*.

  - **Assertion Editor**

    Enables you to create the building blocks of policies. For details, see the *Assertion Editor Guide*.

- **Taxonomy Editor**

  Enables you to create customized taxonomies to categorize your artifacts according to your needs. For details, see the *Taxonomy Editor Guide*.

- **Report Editor**

  Enables you to create customized reports. These can be deployed as a reporting extension and then used in Systinet. For details, see the *Report Editor Guide*.

- **IDE Integrations**

  A set of plug-ins for different development environments offering Systinet integration functionality.

  HP Software provide the following IDE integration products:

  - **Systinet Plug-In for Eclipse**

    Enables you to search the Catalog, generate service clients and skeletons from Systinet resources, perform local resource validation against Systinet policies, and publish local resources to the Catalog. For details, see *Plugin for Eclipse*.

  - **Systinet Plug- In for Visual Studio**

    Enables you to search the Catalog , generate web references from Systinet resources, and publish local resources to the Catalog. For details, see *Plugin for Visual Studio*.

Integrated SystinetProducts

Systinet integrates with the following products:

- **HP SOA Registry Foundation**

  Publish and index your services in a central location. For details, see the *Registry Product Documentation*.

- **HP Business Service Management** (formerly *Business Availability Center*) and **HP Universal Configuration Management Database**

  Discover services stored in UCMDB and access service availability statistics generated by BSM to monitor actual performance against your service level objectives.

- **HP Service Test Management**

  Register your SOAP services in ALM for testing and validation.

UDDI Registry Integration

Systinet enables you to use UDDI Registries as central storage for your service infrastructure or for specific environments such as development and production registries.

Systinet splits responsibility for registry integration between administrators and users with permissions to transfer content between the Catalog and an integrated registry server.

The administrator manages integrated registry servers and must create an integrated server artifact before UDDI integration functionality is available for use. For details, see the following topics in the *Administration Guide*:

- "How to Manage Integrated Products"
- "How to Synchronize Registry Taxonomies"

Caution: If HTTPS is used for Systinet–HP SOA Registry Foundation communication, then it is necessary to import the registry certificates into the application server certificate store. For details, see "SSL Certificates" in the Administration Guide.

Users with appropriate permissions can perform the following content management actions involving an integrated registry server described in the *User Guide*:

- "How to Import from Registry"
- "How to Synchronize with Registries"
- "How to Export to Registry"
- "How to Delete Data from Registry"

A UDDI registry is an implementation of the UDDI specification, for example HP SOA Registry Foundation. The UDDI specification has three major versions, commonly named v1, v2 and v3. Systinet is interoperable with UDDI v3 compliant registries.

The UDDI specification defines four major entities:

- **Business Entity**

  A business entity represents a business unit, company, department, and so on. It contains one or more company names, contacts, and provided services. A business entity in registry corresponds to an Organizational Unit in the Systinet SDM model.

- **Service**

A service represents a logical service. Services cannot stand alone, they must always be part of a parent Business Entity. A service in registry corresponds to an Implementation in the Systinet SDM model.

- **Binding Template**

  A binding template represents technical services. It includes information needed to create and run client applications. A binding template in registry corresponds to an Endpoint in the Systinet SDM model.

- **tModel**

  A tModel represents an arbitrary resource, that cannot be described by the structures above. For example; specification, documentation, (part of) WSDL document, policy or taxonomy. Therefore there is not a common map of tModel to an SDM model artifact.

  Tip: The mapping for certain types of tModel can be defined in Systinet Workbench. For details, see the *Customization Editor Guide*.

The UDDI specification defines interoperable standards for the exchange of data about web services, their interfaces, implementations, deployments, and responsible contacts. For details, see http://uddi.xml.org.

For details about *HP SOA Registry Foundation*, see the Registry Product Documentation.

BSM / UCMDB Integration

HP Business Service Management (formerly Business Availability Center) (BSM) monitors run-time services and collects statistics on their performance. These statistics enable you to verify that a service meets its service level objectives. BSM uses the *Universal Configuration Management Database* (UCMDB) which provides visibility of the infrastructure and applications that support your business services.

Systinet splits responsibility for BSM / UCMDB integration between administrators and users with permissions to access UCMDB content and BSM performance statistics.

The administrator manages integrated BSM / UCMDB servers and must create an integrated server artifact before BSM / UCMDB integration functionality is available for use. For details, see the following topics in the *Administration Guide*:

- "How to Manage Integrated Products"

Note: Systinet can integrate with only one BSM/UCMDB server which is associated with the top-level domain.

Caution: If the Catalog already contains services discovered from BSM/UCMDB, do not delete the BSM/UCMDB server artifact. You may lose access to service discovery functions and discovered artifacts. If your BSM/UCMDB connection settings change, modify the properties of the existing BSM/UCMDB server.

Users with appropriate permissions can perform the following content management actions involving an integrated BSM / UCMDB server described in the *User Guide*:

Systinet artifacts correspond to UCMDB entities as follows:

Systinet Artifact	UCMDB Entity
Organizational unit	Business Unit
Service	Business Service for Catalog
SOAP Service	Web Service

Tip: BSM can access services in Systinet directly if the connection to Systinet is set in *BSM for Systinet Settings*. This enables you to view SOAP Services in the Health Report page. For details, see the *HP Business Service Management* documentation.

ALM Integration

Systinet enables to register the SOAP services in ALM 11 for testing or imports Requirements from ALM 12, and then monitors the testing statistics in Systinet.

To customize Systinet to work with later ALM versions:

1. Access the page Administration -> Configuration -> System Settings.
2. Download the property file: `platform.integration.alm.configuration`
3. Modify this file by adding a new `almServiceFetcher` for the new version of ALM.
4. Upload this file again to Systinet.

Systinet splits responsibility for ALM integration between administrators and users with permissions to register services in ALM and access ALM quality statistics.

The administrator manages integrated ALM servers and must create an integrated server artifact before ALM integration functionality is available for use. For details, see the following topics in the *Administration Guide*:

Note: In order to make Systinet artifacts visible in ALM, you must set the Sharing Principal role to use the `system#everyone` group, share the artifacts you want to make visible in ALM, and set sharing for the appropriate stages of lifecycle processes that govern the artifacts. For details, see

"How to Change the Sharing Principal" in the *Administration Guide*, "How to Share Artifacts" in the *User Guide*, and "How to Define Permissions" in the *Administration Guide*.

Users with appropriate permissions can perform the following content management actions involving an integrated ALM server described in the *User Guide*:

Systinet artifacts correspond to ALM entities as follows:

Systinet Artifact	ALM Entity
SOAP Service	Web Service

Tip: ALM enables you to import services directly from Systinet. For details, see the *HP Application Lifecycle Manager* documentation.

Tip: Systinet provides a set of default technical policies which you can use if you want to make ALM integration and the statistics it provides part of your validation process. For details, see "Default Technical Policies" in the *Administration Guide*.

ALM Server Customizations

You can customize the way that you view data collected from your ALM server by HP Systinet.The HP Systinet server provides three functions to get the service status from the ALM server.

The following procedures list the steps to add and customize your ALM server.

To add the ALM Server:

1. Click **Administration>Servers>ALM Servers>Add ALM Server**. The Add ALM Server page opens.

2. Fill in the following fields:

Name	name of server
Base URL	base URL of server
Username	qcadmin
Password	qcadmin password **Note:** Check the Save Credentials checkbox.
ALM Domain	ALM domain name

ALM Project	ALM project name
Environment	servers environment

3. You can click the **Test Connection** button to test. After the test completes successfully, click **Save**.

To Customize the ALM Server:

1. Click **Customization>Customize** in the left navigation area.
2. Click **Recent Documents>GoogleSearchService** (the SOAP service). The selected SOAP service page opens.
3. Click **More>Metrics**. Initially, you will see a warning that no endpoints are defined for the Testing environment type.
4. Select the **Details** tab, and then under Environments, select **Edit**. The GoogleSearchPort_ Endpoint page opens.
5. In the Environment dropdown menu, select **Testing**, and then click **Save**.
6. Click **More>Metrics** and you will now see the ALM server quality presented graphically by default. However, you can customize this page to display different kinds of information.
7. Click **Customize** to view the **Artifact Detail Page** that you will use to embed your customizations.
8. You can choose to customize with any of the following scripts:

- Gets all ALM servers as JSON array by getAlmServers() function:

```
<customization xmlns="http://systinet.hp.com/2009/02/ui/customization"
xmlns:cust="http://systinet.hp.com/2009/02/ui/customization"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="cust artifactDetail.xsd">

<content>

<server id="alm_integration_server">

<import location="/scripts/almServerAPI.js"/>

</server>

<html id="alm_integration_client">

<include>

<script>

Ext.onReady(function () {

getAlmServers(almServersListener);
function almServersListener(result){

alert(result);

}

});

</script>

</include>

</html>

</content>

</customization>
```

- Gets all ALM beans from ALM server as JSON objects by getAlmBeans(serverUUID, beanProps, runtimeContext) function. Includes:

Name	Description
serverUUID	The ALM server UUID.
beanProps	The name and view name (config in property file of 'platform.integration.alm.configuration' above) of ALM bean as JSON array. For example, var beanProps = [{name:almReq, href:viewRequirement}, {name:almDefect, href:viewDefect}]
runtimeContext	The runtime context to get ALM bean, must include serviceUUID. For example, var runtimeContext = {serviceUUID:alm_integration_client.uuid}

```
<customization xmlns="http://systinet.hp.com/2009/02/ui/customization"
xmlns:cust="http://systinet.hp.com/2009/02/ui/customization"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="cust artifactDetail.xsd">

<content>

<server id="alm_integration_server">

<import location="/scripts/almServerAPI.js"/>

</server>

<html id="alm_integration_client">

<parameter name="uuid">${artifact._uuid}</parameter>

<include>

<script>

Ext.onReady(function () {

var serverUUID = "xxx";
var beanProps = "[{name:almReq, href:viewRequirement}, {name:almDefect,
href:viewDefect}, {name:almTest, href:viewTest}]";

var runtimeContext = {serviceUUID:alm_integration_client.uuid};

getAlmBeans(serverUUID, beanProps, runtimeContext, almBeansListener);

function almBeansListener(result){

alert(result);

}

});

</script>

</include>

</html>

</content>

</customization>
```

- Gets all ALM beans from all ALM server as JSON object by getServiceStatus(beanProps, runtimeContext) function. Includes:

Name	Description
beanProps	The name and view name (config in property file of 'platform.integration.alm.configuration' above) of ALM bean as JSON array. For example, var beanProps = [{name:almReq, href:viewRequirement}, {name:almDefect, href:viewDefect}]
runtimeContext	The runtime context to get ALM bean, must include serviceUUID. For example, var runtimeContext = {serviceUUID:alm_integration_client.uuid}

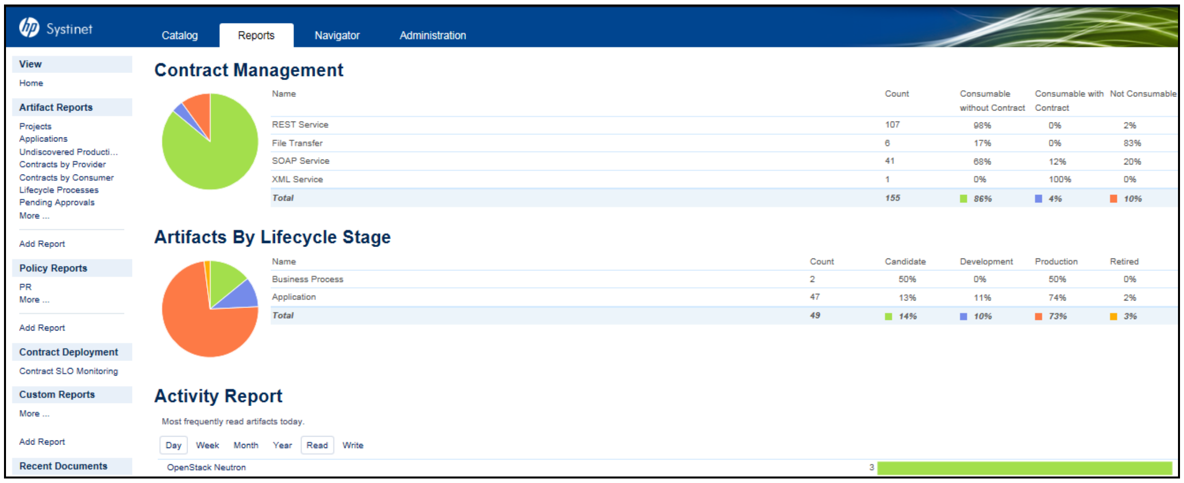
```
<customization xmlns="http://systinet.hp.com/2009/02/ui/customization"
xmlns:cust="http://systinet.hp.com/2009/02/ui/customization"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="cust artifactDetail.xsd">

<content>

<server id="alm_integration_server">

<import location="/scripts/almServerAPI.js"/>

</server>

<html id="alm_integration_client">

<parameter name="uuid">${artifact._uuid}</parameter>

<include>

<script>
```

```
Ext.onReady(function () {

var beanProps = "[{name:almReq, href:viewRequirement}, {name:almDefect,
href:viewDefect}, {name:almTest, href:viewTest}]";

var runtimeContext = {serviceUUID:alm_integration_client.uuid};

getServiceStatus(beanProps, runtimeContext, serviceStatusListener);

function serviceStatusListener(result){

alert(result);

}

});

</script>

</include>

</html>

</content>

</customization>
```

Chapter 13: Reporting

The initial page of the reporting module presents three predefined reports based on contract management, artifacts by lifecycle stage and an activity report. Clicking on contract types or lifecycles stages alters the pie chart to reflect the selection. You may also add relevant and interesting feeds to this page (see in User Guide).

Screenshot: Report Tab



Contract Management

Name	Count	Consumable without Contract	Consumable with Contract	Not Consumable
REST Service	107	98%	0%	2%
File Transfer	6	17%	0%	83%
SOAP Service	41	68%	12%	20%
XML Service	1	0%	100%	0%
Total	*155*	🟩 *86%*	🟦 *4%*	🟧 *10%*

Artifacts By Lifecycle Stage

Name	Count	Candidate	Development	Production	Retired
Business Process	2	50%	0%	50%	0%
Application	47	13%	11%	74%	2%
Total	*49*	🟩 *14%*	🟦 *10%*	🟧 *73%*	🟨 *3%*

Activity Report

Most frequently read artifacts today.

Day Week Month Year Read Write

OpenStack Neutron	3

- **Contract Management**
  - Display of specific artifact types, number of artifact instances, number of consumable instances without/with contract, and number of non-consumable artifacts.
  - Administrator can customize which artifact types are captured in these reports.
  - The graph is for a selected line and can be changed by selecting the a different line from the report.
- **Lifecycle**
  - Displays the distribution of artifact instances between lifecycle stages.
  - Administrator can define which artifact types and which lifecycle stages are displayed.
  - The graph is for a selected line and can be changed by selecting the a different line from the report.
- **Users Activity**
  - Displays most frequently read or written artifacts over a specific time period (day/week or month).

Note: The reports on this page can only be modified by users with administrator rights.

There are three report types available in Systinet:

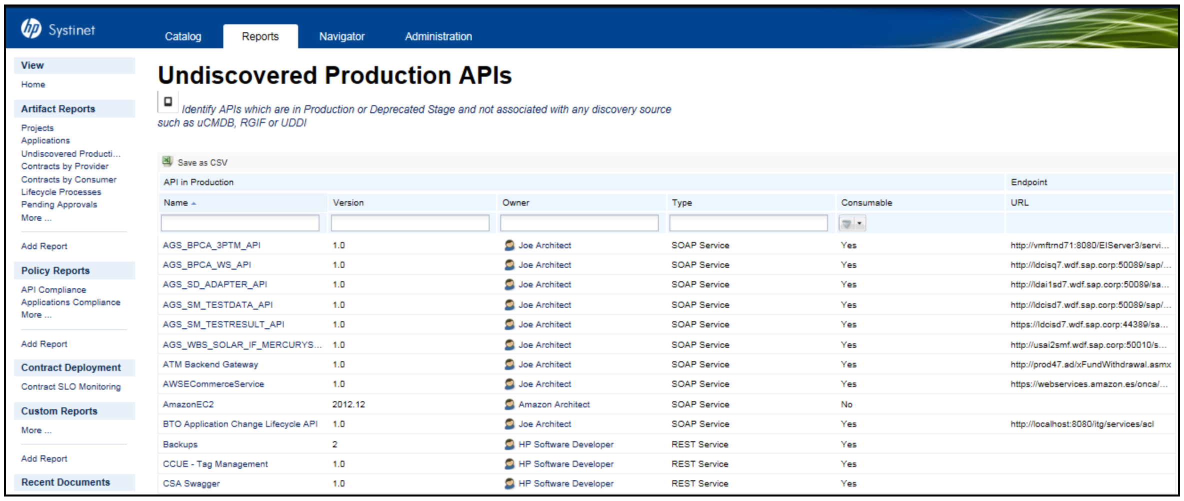
- "Artifact Reports" below
- "Policy Reports" on the next page
- "Custom Reports" on page 87

Artifact Reports

Artifact reports are tabular reports based on DQL. Users can use them together with powerful table capabilities, such as filtering or sorting, in order to present information that cannot be returned via the Search form.

For example, the report *Services by Implementations* lists all services together with their providers, their implementations and implementation endpoints. A user can easily find all the Services from a specific provider that have Implementation in the production and Endpoint of specific type. Alternately, they can find all the Services from a specific domain that have a JMS endpoint.

The Artifact Reports menu contains a default set of reports which can be customized. See "How to Add Reports to the Menu" in the *User Guide* for more details.



Undiscovered Production APIs

Identify APIs which are in Production or Deprecated Stage and not associated with any discovery source such as uCMDB, RGIF or UDDI

API in Production					Endpoint
Name	Version	Owner	Type	Consumable	URL
AGS_BPCA_3PTM_API	1.0	Joe Architect	SOAP Service	Yes	http://vmftrnd71:8080/EIServer3/servi...
AGS_BPCA_WS_API	1.0	Joe Architect	SOAP Service	Yes	http://ldcisq7.wdf.sap.corp:50089/sap/...
AGS_SD_ADAPTER_API	1.0	Joe Architect	SOAP Service	Yes	http://ldai1sd7.wdf.sap.corp:50089/sa...
AGS_SM_TESTDATA_API	1.0	Joe Architect	SOAP Service	Yes	http://ldcisd7.wdf.sap.corp:50089/sap/...
AGS_SM_TESTRESULT_API	1.0	Joe Architect	SOAP Service	Yes	https://ldcisd7.wdf.sap.corp:44389/sa...
AGS_WBS_SOLAR_IF_MERCURYS...	1.0	Joe Architect	SOAP Service	Yes	http://usai2smf.wdf.sap.corp:50010/s...
ATM Backend Gateway	1.0	Joe Architect	SOAP Service	Yes	http://prod47.ad/xFundWithdrawal.asmx
AWSECommerceService	1.0	Joe Architect	SOAP Service	Yes	https://webservices.amazon.es/onca/...
AmazonEC2	2012.12	Amazon Architect	SOAP Service	No	
BTO Application Change Lifecycle API	1.0	Joe Architect	SOAP Service	Yes	http://localhost:8080/itg/services/acl
Backups	2	HP Software Developer	REST Service	Yes	
CCUE - Tag Management	1.0	HP Software Developer	REST Service	Yes	
CSA Swagger	1.0	HP Software Developer	REST Service	Yes	

Content	Description
Provider name	Name of the provider.
Email	Email address of the provider.
API in Production Name	Name of Applications during production.

Version	The version number of the artifact.
Type	Type of services.
Consumable	Select to make the XML service available to consumers.
Endpoint URL	URL for the actual implementation of the service.

Policy Reports

Policy Reports display the compliance of artifacts selected by a repository search (or from external documents) against a set of technical policies. They can be used, for example, when Architects want to check the compliance of all WSDL documents containing a specific keyword against specified technical policies (e.g. WS-I Basic Profile Policy). Only policies defined in the current policy report are considered and eventual policies for the current lifecycle stage are ignored. Policy reports are also not affected by artifact compliance status calculations.

Policy Reports are recalculated automatically. By default they are refreshed daily, but Administrators can change the timing by scheduling an appropriate built-in task. The system can also recalculate reports based on a user's demand, but recalculation may be time consuming.

Systinet provides 2 new OOTB Policy Reports:

1. API Compliance Report

2. Application Compliance Report

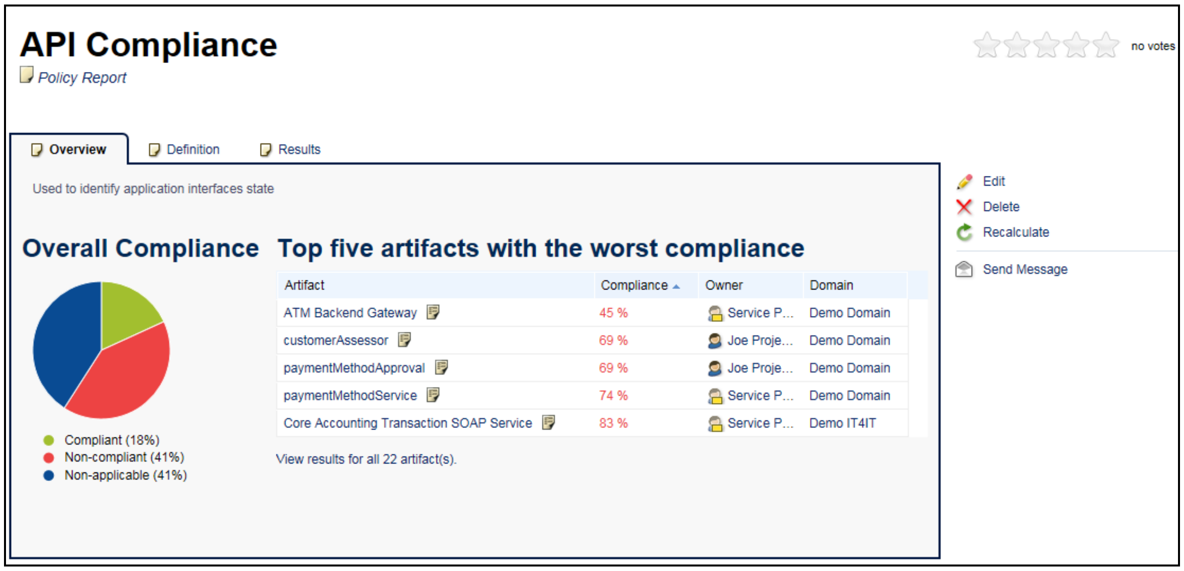
API Compliance Report

This report is used to identify the application interfaces state. There are two ways to access the API Compliance Report page:

1. Go to **Catalog** homepage->**Manage APIs**->**API Compliance** or

2. Go to **Reports tab** homepage-> **Policy Reports** section in left menu > select the report API Compliance.

This displays the API Compliance homepage.



API Compliance

Policy Report

no votes

Overview | Definition | Results

Used to identify application interfaces state

Edit
Delete
Recalculate
Send Message

Overall Compliance

● Compliant (18%)
● Non-compliant (41%)
● Non-applicable (41%)

Top five artifacts with the worst compliance

Artifact	Compliance ▲	Owner	Domain
ATM Backend Gateway	45 %	Service P...	Demo Domain
customerAssessor	69 %	Joe Proje...	Demo Domain
paymentMethodApproval	69 %	Joe Proje...	Demo Domain
paymentMethodService	74 %	Service P...	Demo Domain
Core Accounting Transaction SOAP Service	83 %	Service P...	Demo IT4IT

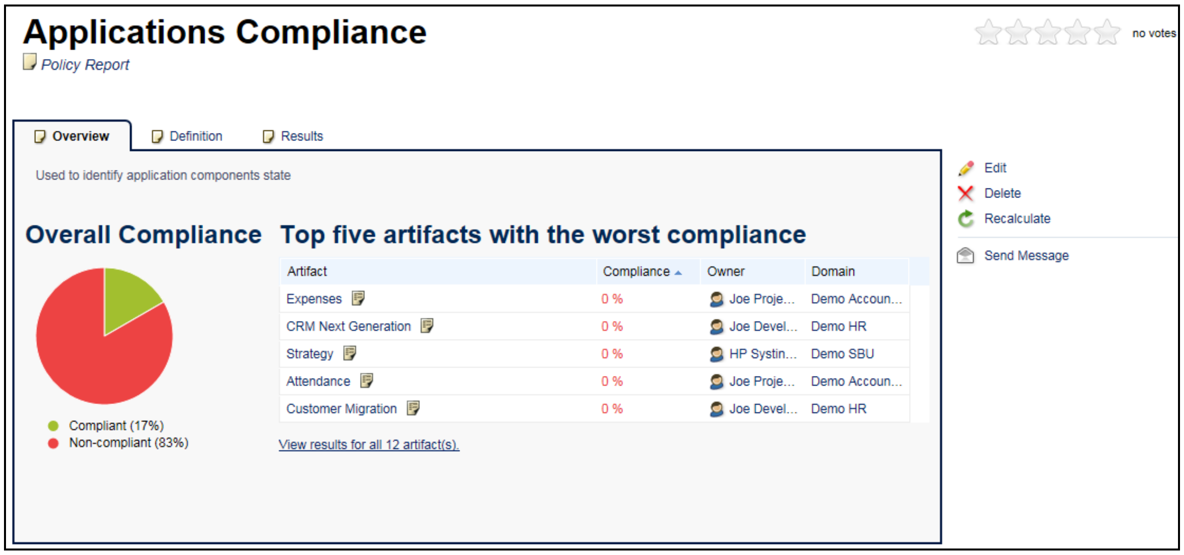
View results for all 22 artifact(s).

Application Compliance Report

This report is used to identify the application components state. There are two ways to access the Application Compliance Report page:

1. Go to **Catalog** homepage->**Applications**->**Application Compliance** or
2. Go to **Reports tab** homepage-> **Policy Reports** section in left menu > select the report Application Compliance.

This displays the Application Compliance homepage.



Applications Compliance

Policy Report

no votes

Overview | Definition | Results

Used to identify application components state

Edit
Delete
Recalculate
Send Message

Overall Compliance

● Compliant (17%)
● Non-compliant (83%)

Top five artifacts with the worst compliance

Artifact	Compliance ▲	Owner	Domain
Expenses	0 %	Joe Proje...	Demo Accoun...
CRM Next Generation	0 %	Joe Devel...	Demo HR
Strategy	0 %	HP Systin...	Demo SBU
Attendance	0 %	Joe Proje...	Demo Accoun...
Customer Migration	0 %	Joe Devel...	Demo HR

View results for all 12 artifact(s).

Custom Reports

Custom Reports are based on report definitions as stored in the Reporting Server. These reports can combine multiple data sources. For example, it is possible to mix data from the repository with data from an external database.

While Custom Report definitions are stored in the Reporting Server and are managed with the Report Editor, artifact reports are customizable directly from the web UI in a user-friendly way.

Custom Reports are constructed in Report Editor and are available in the **Reports** tab under **Custom Reports** in the left-hand menu. For details, see *Report Editor Guide* and "How to Add Custom Reports" in the *User Guide*.

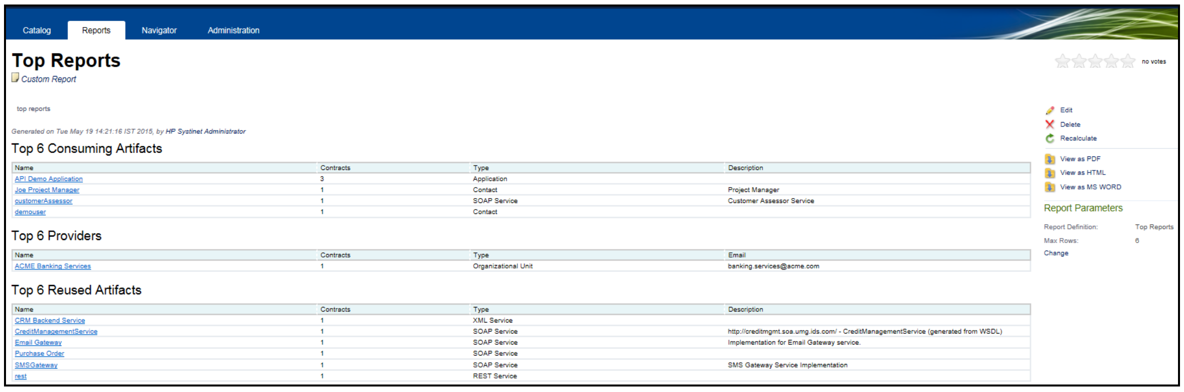
Systinet provides 2 OOTB Custom Reports:

1. Top Reports
2. API Documentation

Top Reports

This report displays the most active providers, frequently consumed and often reused artifacts.

1. Select **Top Reports** from the **Report Definition** dropdown box.
2. Enter a name for the new report and an optional text describing the report.
3. Define a maximum number of rows for the report.
4. Click **Save** to generate the report.



Top Reports

Custom Report

top reports

Generated on Tue May 19 14:21:16 IST 2015, by HP Systinet Administrator

Top 6 Consuming Artifacts

Name	Contracts	Type	Description
API Demo Application	3	Application	
Joe Project Manager	1	Contact	Project Manager
customerAssessor	1	SOAP Service	Customer Assessor Service
demouser	1	Contact	

Top 6 Providers

Name	Contracts	Type	Email
ACME Banking Services	1	Organizational Unit	banking.services@acme.com

Top 6 Reused Artifacts

Name	Contracts	Type	Description
CRM Backend Service	1	XML Service	
CreditManagementService	1	SOAP Service	http://creditmgmt.soa.umg.ids.com/ - CreditManagementService (generated from WSDL)
Email Gateway	1	SOAP Service	Implementation for Email Gateway service.
Purchase Order	1	SOAP Service	
SMSGateway	1	SOAP Service	SMS Gateway Service Implementation
rest	1	REST Service	

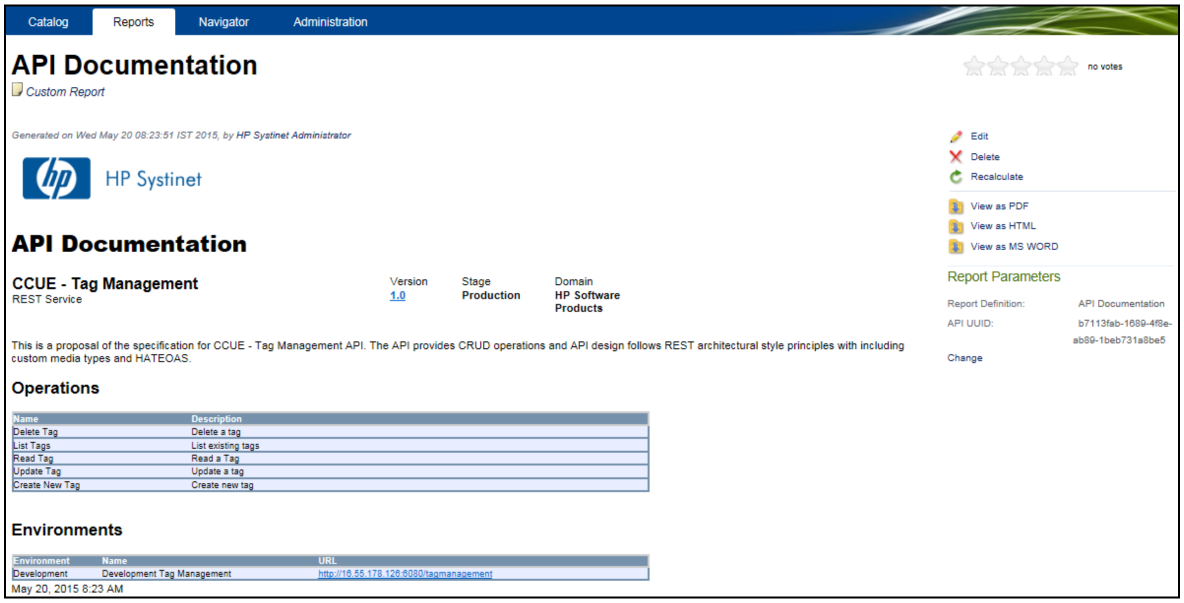
Report Parameters

Report Definition:	Top Reports
Max Rows:	6

API Documentation

This report provides information on the application interfaces artifact (for example, REST and SOAP services) using the API UUID.

1. Select **API Documentation** from the **Report Definition** dropdown box.
2. Enter a name for the new report and an optional text describing the report.
3. Enter the API UUID from the created artifact.
4. Click **Save** to generate the report.



API Documentation

Custom Report

Generated on Wed May 20 08:23:51 IST 2015, by HP Systinet Administrator

HP Systinet

API Documentation

CCUE - Tag Management
REST Service

Version	Stage	Domain
1.0	Production	HP Software Products

This is a proposal of the specification for CCUE - Tag Management API. The API provides CRUD operations and API design follows REST architectural style principles with including custom media types and HATEOAS.

Operations

Name	Description
Delete Tag	Delete a tag
List Tags	List existing tags
Read Tag	Read a Tag
Update Tag	Update a tag
Create New Tag	Create new tag

Environments

Environment	Name	URL
Development	Development Tag Management	http://16.55.178.126:6080/tagmanagement

May 20, 2015 8:23 AM

Report Parameters

Report Definition:	API Documentation
API UUID:	b7113fab-1689-4f8e-ab89-1beb731a8be5

Note: Top reports and API documentation reports can be viewed as PDF and HTML by clicking the links on the right menu.

Chapter 14: Administration

Administration in HP Systinet can be broadly divided into the following areas:

- **Managing Content**

  The most important content management concept in Systinet is the use of *Domains*. The administrator can create a domain structure that represents your organizational structure. Each domain represents a working area with users assigned to specific roles within each domain and the content of the domain managed to restrict its visibility and access rights. For more details, see "Domains" on page 92.

  The administrator is also responsible for the day-to-day maintenance of the data content in the Catalog and reports about its status. Systinet provides a set of administration tasks that the administrator can execute manually or schedule to run at set times or periodically to maintain and update this information. For details, see "Administration Task Management" in the *Administration Guide*.

- **Managing Users**

  The management of users is normally delegated to an external user store, such as LDAP, where the management of the people and groups who actually use Systinet should take place. Systinet represents users with *User* artifacts which the particular user or an administrator can manage. Users can create additional *Contact* artifacts to represent external contacts who do not use the product in order to associate them with particular artifacts in the Catalog. For details, see "User Management" in the *Administration Guide*.

  If necessary, the administrator can add and manage additional local groups to organize users into groups that are not represented by the external user store. For details, see "Group Management" in the *Administration Guide*. HP recommends using roles instead of creating local groups.

  An important concept in Systinet is the use of *Roles*. Roles are generic job descriptions that can apply to users and groups in specific domains. The use of roles enables the administrator to manage generic templates for lifecycle processes and security management in the top-level global domain which the resolve to specific users and groups within each working domain. Roles also control user access to functionality in the user interface. For more details, see "Roles" on page 96.

  Administrators within each domain are responsible for assigning users to roles within their domain. For details, see "Domains" on page 92.

- **Managing Security**

  The administrator is responsible for managing and controlling user access to Catalog content. Systinet uses Access Control Lists (ACL) to restrict access based on users, groups, or roles. For details, see "Security and Access Control" on the next page.

- **Managing Global Artifacts**

  Systinet uses domains to divide content into working areas with users assigned to specific roles within the domain. Containing all the working domains is a top-level domain which contains global artifacts which apply across all domains. Systinet restricts access to these artifacts and their management to the top-level administrator. For details, see "Lifecycle Management" and "Policy Management" in the *Administration Guide*.

- **Configuration and System Management**

  The administrator is responsible for the configuration of each deployment of Systinet.

  The Administration tab provides access to certain aspects of the configuration which can be managed while Systinet is running. For details, see "Configuration Management" in the *Administration Guide*.

  The administrator can customize the user interface. For details, see "UI Customization Overview" in the *Administration Guide*.

  Certain aspects of the configuration are not accessible during runtime. Systinet provides a set of command-line tools and applications that enable this configuration. For details, see "Administration Utilities" in the *Administration Guide*.

Security and Access Control

Most organizations restrict access to resources by user and group permissions. Systinet extends this type of security by enabling the use of domain and role-based access rights.

Systinet uses Access Control Lists (ACL) to define who can access particular resources and their permissions. Each ACL consists of a set of Access Control Elements (ACE) which define the following for a resource or collection of resources:

- **User Identification**

  The user identification as a specified user, a group of users, or a role that resolves to users and groups in the domain that the artifact belongs to.

- **Granted Permission**

  One of the following:

  - **Read Permission**

    Access to read the data and metadata of an artifact or resource, or a collection of artifacts.

  - **Write Permission**

Access to modify the data and metadata of an artifact or resource, or to create new artifacts, resources, and sub-collections, and update the metadata of a collection of artifacts. Users assigned as the owner of an artifact and administrators always have write permission.

ACLs apply in the following use cases:

- **Artifact Creation Rights**

  The administrator can define which roles can create artifact types within a domain. Within the domain, the users in the allowed roles can access the artifact creation pages for the specified artifact types. The default creation rights are cumulative, so default rights given in the top-level domain apply in all other domains, and rights given to a group or role also apply in addition to rights given to each user in the group or role. For details, see "How to Manage Default Access Rights" on page 1.

- **Governed Artifact Access Rights**

  The access rights for artifacts in governance are determined by the lifecycle process applicable to the artifact. The administrator can assign rights and permissions to particular roles for each stage of a lifecycle process. Within a domain, these roles resolve to the assigned users and groups who have the specified access to the artifact at that lifecycle stage. For details, see "How to Define Permissions" on page 1.

- **Ungoverned Default Artifact Access Rights**

  In the cases where artifacts are not governed, the administrator can define which roles can read or write particular artifact types within a domain. Within the domain, the users in the allowed roles can access the artifact edit pages for the specified artifact types. In addition, you can extend this default access control functionality using particular values of categorization properties. For example, this enables you to define different access rights for services categorized as application services and for services categorized as infrastructure services. The default access rights are cumulative, so default rights given in the top-level domain apply in all other domains, and rights given to a group also apply to all the users of the group. For details, see "How to Manage Default Access Rights" on page 1.

- **Specific Artifact Access Rights**

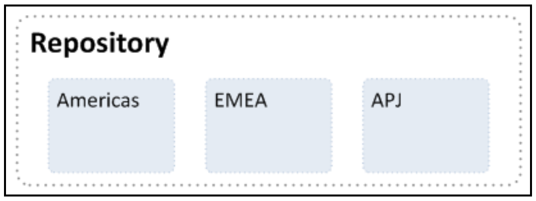
  The owner, maintainers, and administrators of artifacts can set read or write access rights to artifacts for users, groups, or roles individually or in bulk. For details, see "How to Edit Access Rights" in the *User Guide*.

  Caution: HP recommends only setting specific artifact access rights for ungoverned artifacts.

Chapter 15: Domains

Domains provide a logical separation of data within the Catalog. Each domain can represent a discrete working area for an individual department or organizational unit. This separation allows users to focus on the data that is most relevant to them and enables data to be structured by working area.

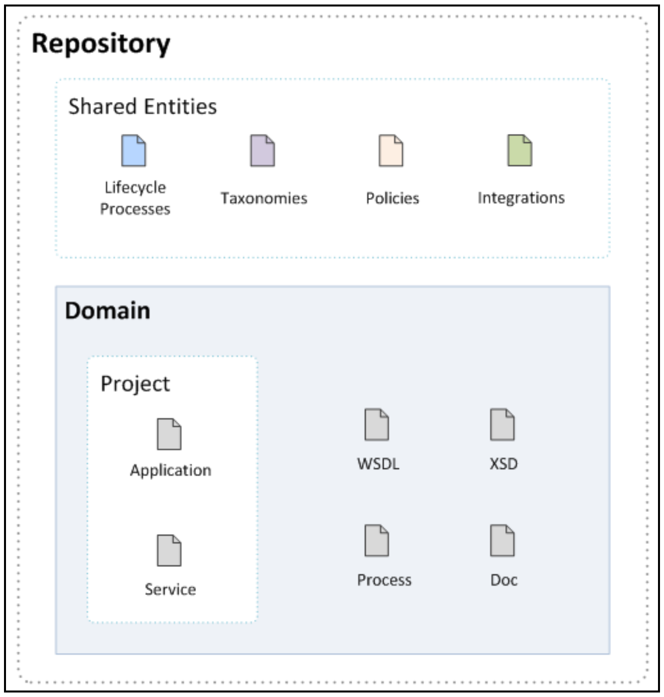
In this release, Systinet provides support for a single layer of domains within a global top-level repository domain. For example, a domain structure representing organizational regions, Americas, EMEA, and APJ.



After installation, Systinet consists of the top-level repository domain and a default domain. The default domain represents a default working area for all users until the administrator creates additional working area domains and assigns users to them.

Note: The administrator can disable default domain sign-in, so that users must be assigned to a specific domain before they can sign-in. For details, see "Disabling Default Domain Sign-In" on page 1.

The top-level domain is a special domain containing system and global data, such as lifecycle processes, policies, and taxonomies, which apply across all domains, and each working domain contains the specific data relevant to users of that domain.



The exact separation of data between the top-level and working domains is as follows:

Top-Level Domain (Global Configuration)

- Lifecycle Processes
- Taxonomies
- Policies
- Roles Definition
- BSM servers
- System settings (including SDM and UI customizations)

Working Domains

- Artifacts (for example, Services, Applications, WSDLs, and Documents) that belong to the domain
- User Role Assignments (for example, Joe is an architect in the EMEA domain)
- Default Settings (for example, a default server folder)
- UDDI Registry / ALM Servers

Working domains inherit all settings applied in the top-level repository domain. For example, access rights, roles, and lifecycle processes set in the top-level domain apply in all domains.

This domain structure creates a logical separation, not only between departments or organizational units, but also between global functions and working area domain functions.

Users in Systinet perform specific functionality based on the roles assigned to them and the user interface restricts their access to functionality and artifacts based in these roles.

The user roles are split into the following user types:

- **Top-Level Repository Administrators**

  Global administration with responsibility for the following functional areas:

  - Domain Management for all domains. For details, see "Domain Management" on page 1.
  - Lifecycle process Administration. For details, see "Lifecycle Management" in the *Administration Guide*.
  - Integrated Products Server administration for BSM. For details, see "Product Integration" on page 71.
  - User and Group Management. For details, see "User Management" and "Group Management" in the *Administration Guide*.
  - Role Administration. For details, see "Role Management" in the *Administration Guide*.
  - Server Configuration Management. For details, see "Configuration Management" in the *Administration Guide*.
  - Policy Administration. For details, see "Policy Management" on page 103.
  - The Administrator can also access all the functionality of Domain Administrators.

- **Domain Administrators**

  Users assigned to the administrator role in a specific domain with responsibility for the following functional areas:

  - Domain Management for their domain. For details, see "Domain Management" in the *Administration Guide*.
  - Management of administrative tasks within their domain. For details, see "Administration Task Management" in the *Administration Guide*.
  - Integrated Products Server administration for UDDI Registries and ALM. For details, see "Product Integration" on page 71.

- **Domain Users**

  Users assigned to a specific role within a domain with specific functionality associated with that role. The same user can access different domains in different roles.

This separation of functions and roles is described in more detail in "Roles" on page 96.

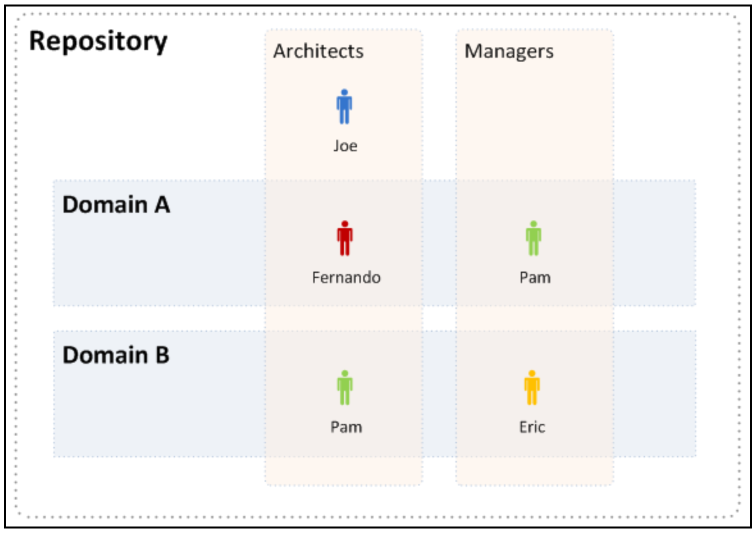
Each artifact belongs to exactly one domain. This domain is set to the current domain when a user creates an artifact. Typically, the domain does not change during the artifact lifecycle, but if required it is possible to transfer single artifact or multiple artifacts from one domain to another one using the Change Domain operation. For details, see "How to Edit Artifact Domains" in the *User Guide*.

By default, artifacts are only visible in the owning domain but they can be explicitly shared for all users across all domains using the Share operation. For details, see "How to Share Artifacts" in the *User Guide*. Typically, this operation applies to artifacts entering production and associated with a lifecycle process. For details, see "How to Define Automatic Actions" in the *Administration Guide*.

Chapter 16: Roles

Systinet offers functionality across the entire service development lifecycle. In most organizations, these functions are performed by many individuals and teams with specific permissions. Systinet uses *Roles* to enable you to define and assign these permissions, and use these assignments to focus each user or group on specific functionality and tasks and restrict their access to artifacts appropriate to their role.

The administrator defines roles in the top-level domain, but user assignment to roles can be global or to different roles in different domains. For example, in the following diagram, Joe is assigned a global architect role in the top-level repository domain, Fernando is assigned the architect role in Domain A, Pam is assigned the manager role in Domain A and the architect role in Domain B, and Eric is assigned to the manager role in Domain B.



These assignments mean that in Domain A, Joe and Fernando access functionality and artifacts relevant to the architect role, whereas Pam accesses functionality relevant to the manager role.

Pam has a different role in Domain B, so along with Joe, accesses architect functionality, whereas Eric accesses functionality relevant to the manager role.

To assign users or groups to roles, see "How to Manage User Roles in Domains" in the *Administration Guide*. For more details about domains, see "Domains" on page 92.

The following topics describe in more detail how Systinet uses roles:

- "Roles in the User Interface" on the next page

  The UI uses roles to restrict the availability of functionality to users in appropriate roles.

- "Roles in Lifecycle" on page 98

You can create Lifecycle templates with specified tasks and actions assigned to specific roles.

- "Security and Access Control" on page 90

  Systinet restricts access to artifact types using ACLs which can use roles as well as users and groups.

The default roles and their assigned functionality are described in the following topics:

- "Business Partner Role" on page 99
- "Business Analyst Role" on page 100
- "Service Provider Role" on page 100
- "Administrator Role" on page 101

The administrator can extend the default roles by adding additional customized roles. For details, see "How to Manage Roles" in the *Administration Guide*.

Systinet also includes a special role, Sharing Principal, specifically associated with sharing artifacts. By default, this role is associated with the `system#registered` group which represents all users who access Systinet. For more details, see "How to Change the Sharing Principal" in the *Administration Guide*.

Roles in the User Interface

Systinet restricts access to UI functionality according to your role.

- **Business Partner**

  The Catalog tab provides search and browse functionality for consumable artifacts and enables users in the Business Partner role to request contracts. For more details, see "Business Partner Role" on page 99.

  The Navigator tab provides a visual representation of the relationships between artifacts. It provides various layouts and role-based filters to enable you to visualize the content of the repository relevant to your role. For more details, see "Navigator" on page 20.

- **Business Analyst**

  The Catalog tab for Business Analysts extends Business Partner functionality. It enables users in the Business Analyst role to search and browse service-related artifacts, to interact with lifecycle, and to create service-related artifacts. For more details, see "Business Analyst Role" on page 100.

  The Navigator tab provides a visual representation of the relationships between artifacts. It provides various layouts and role-based filters to enable you to visualize the content of the repository relevant to your role. For more details, see "Navigator" on page 20.

- **Service Provider**

  The Catalog tab for Service Providers extends Business Analyst functionality. It enables users in the Service Provider role to create, develop, and manage service-related artifacts. For more details, see "Service Provider Role" on page 100.

  The Navigator tab provides a visual representation of the relationships between artifacts. It provides various layouts and role-based filters to enable you to visualize the content of the repository relevant to your role. For more details, see "Navigator" on page 20.

  The Reports tab provides access to view and create reports about Catalog content. For more details, see "Reporting" on page 83.

- **Administrator**

  In addition to all the tabs and functionality accessible by users in the Service Provider role, administrators access an Administration tab to enable them to manage users, groups, roles, domains, and other system artifacts. For more details, see "Administration" on page 89.

Roles in Lifecycle

Lifecycle management makes use of roles to enable global lifecycle process management with role-based assignments in a lifecycle process template.

The administrator of the top-level domain manages lifecycle processes and uses roles to define the following:

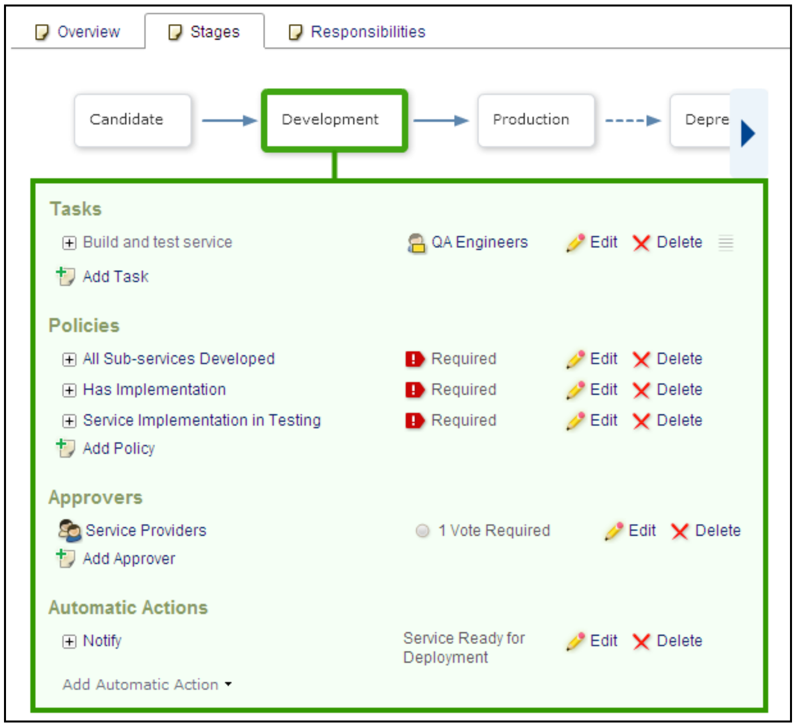
- The user role responsible for approving particular lifecycle stages.
- The user roles with read-only, write access, and ownership permissions for artifacts at particular lifecycle stages.
- The user roles responsible for performing tasks associated with a lifecycle stage.
- The user roles that are automatically notified as a result of specific events within the lifecycle.

For more details about lifecycle governance, see "Lifecycle" on page 61.

Within specific domains, these role assignments resolve to the users and groups assigned to the role in that domain. For example, consider the development stage of a lifecycle process for services.

The stage might consist of the following role assignments:

- A task to build and test the service assigned to the QA Engineer role.
- A stage approver in the Service Provider role.
- An automatic action to notify the users in the Operations Manager role when a service is approved at the development stage.



Tasks

Build and test service	QA Engineers	Edit	Delete
Add Task			

Policies

All Sub-services Developed	Required	Edit	Delete
Has Implementation	Required	Edit	Delete
Service Implementation in Testing	Required	Edit	Delete
Add Policy			

Approvers

Service Providers	1 Vote Required	Edit	Delete
Add Approver			

Automatic Actions

Notify	Service Ready for Deployment	Edit	Delete
Add Automatic Action			

In different domains, different users perform each of these roles. Within domains, Systinet replaces the roles in the template with the specific users and groups assigned to that role in that domain.

In the EMEA domain, this could result in the following specific assignments:

- A member of the EMEA QA Engineers group, assigned to the QA Engineer role, must complete the Build and Test task.
- An EMEA domain user, assigned to the Service Provider role, must approve the development stage for the service.
- When the service is approved, all users in the Operations Manager role in the EMEA domain are notified that the service is ready for deployment.

In the US domain these assignments are to different users and groups performing the same roles.

Business Partner Role

Business Partner is a consumer role, typically an external business-to-business user, group or a company. Typically, service consumers are seeking to reduce time to market and reduce the cost of the development. They build composite applications by searching for existing services and negotiating contracts with service providers.

Users in the Business Partner role perform the following functions:

- Search and Discover consumable service artifacts. For details, see "Search and Browse" on page 15.
- Visualize the Catalog content. For details, see "Navigator" on page 20.
- Collaborate with your colleagues and service providers. For details, see "Collaboration" on page 26.
- Request contracts to consume service artifacts. For details, see "Contracts" on page 50.

Business Analyst Role

A Business Analyst is a negotiator between the business side of an enterprise and the providers of services to the enterprise. The business analyst understands business problems and opportunities in the context of the requirements and recommends solutions that enable the organization to achieve its goals. Solutions often include a systems development component, but may also consist of process improvement or organizational change.

Users in the Business Analyst role perform the following functions:

- Search and Discover consumable service artifacts. For details, see "Search and Browse" on page 15.
- Visualize the Catalog content. For details, see "Navigator" on page 20.
- Collaborate with your colleagues and service providers. For details, see "Collaboration" on page 26.
- Author new Catalog content. For details see "Authoring" on page 27.
- Edit existing Catalog content. For details, see "Artifact Management" on page 41.
- Create new versions of existing Catalog content. For details, see "Versioning" on page 47.
- Request, approve and revoke contracts to consume service artifacts. For details, see "Contracts" on page 50.
- Participate in the lifecycle of Catalog content. For details, see "Lifecycle" on page 61.

Service Provider Role

The *Service Provider* creates, develops, manages, and provides services, applications, and processes. In contrast to Business Partner they are interested in the design-time policies, test cases, functional specifications, services in different stages, and reporting tools.

Users in the Service Provider role perform the following functions:

- Search and Discover consumable service artifacts. For details, see "Search and Browse" on page 15.
- Visualize the Catalog content. For details, see "Navigator" on page 20.
- Collaborate with your colleagues and service providers. For details, see "Collaboration" on page 26.
- Author new Catalog content. For details see "Authoring" on page 27.
- Edit existing Catalog content. For details, see "Artifact Management" on page 41.
- Create new versions of existing Catalog content. For details, see "Versioning" on page 47.
- Request, approve and revoke contracts to consume service artifacts. For details, see "Contracts" on page 50.
- Participate in the lifecycle of Catalog content. For details, see "Lifecycle" on page 61.
- View and create reports on Catalog content. For details, see "Reporting" on page 83.
- Export, import, synchronize Catalog content with integrated products and review the data they return. For details, see "Product Integration" on page 71.

Administrator Role

Systinet provides an administrator role with responsibility for managing users, groups, roles, and system artifacts. The responsibilities of administrators vary according to the domain that they manage.

- **Top-Level Repository Administrators**

  Global administration with responsibility for the following functional areas:

  - Domain Management for all domains. For details, see "Domain Management" in the *Administration Guide*.
  - Lifecycle Process Administration. For details, see "Lifecycle Management" in the *Administration Guide*.
  - Integrated Products Server administration for BSM. For details, see "Product Integration" on page 71.
  - User and Group Management. For details, see "User Management" and "Group Management" in the *Administration Guide*.
  - Role Administration. For details, see "Role Management" in the *Administration Guide*.
  - Server Configuration Management. For details, see "Configuration Management" in the *Administration Guide*.

- Policy Administration. For details, "Policy Management" on page 103.
- The Administrator can also access all the functionality of Domain Administrators.

- **Domain Administrators**

  Users assigned to the administrator role in a specific domain with responsibility for the following functional areas:

  - Domain Management for their domain. For details, see "Domain Management" in the *Administration Guide*.
  - Management of administrative tasks within their domain. For details, see "Administration Task Management" in the *Administration Guide*.
  - Integrated Products Server administration for UDDI Registries and ALM. For details, see "Product Integration" on page 71.

Chapter 17: Policy Management

Systinet enables you to validate your Catalog content against published policies to ensure its consistency and conformance to your business policy.

Policy management and validation uses the following artifact types:

- **Technical Policy**

  A technical policy consists of a set of assertions and references to other technical policies, and serves as the central point of reference for validation. References to other technical policies enable you to collect a set of technical policies together into a larger policy enabling you to validate them collectively.

- **Assertion**

  An assertion is a validation check for a single piece of data which can either pass or fail. For example, an assertion can check the following attributes of an artifact:

  - **Property Values**

    An assertion can check that a particular property has a value and what the value is. Typical examples include verifying that keywords are set when an artifact is created and verifying that an artifact is consumable before it enters the production lifecycle stage.

  - **Related Artifacts**

    An assertion can check whether a particular artifact type is associated with the artifact being verified and check property values of the related artifacts. A typical example is to verify that a service has an attached document which is categorized as a business specification document before the service is approved at the candidate lifecycle stage.

  - **Data Content**

    Some artifact types are expected to contain attached content. An assertion can verify that the attachment exists.

  - **Artifact State**

    An assertion can check various status attributes of an artifact, For example, its lifecycle status or its compliance status.

  - **Metrics from Integrated Products**

    An assertion can check values generated by integrated products. For example, whether there are any open defects in *HP Application Lifecycle Manager* (ALM).

  Create and edit assertions using Assertion Editor. For details, see the *Assertion Editor Guide*.

Systinet enables you to use technical policies in the following ways:

- **Lifecycle Stage Validation**

  Lifecycle stages in a lifecycle process can have a set of policies associated with them that serve as validation checks for artifacts at that lifecycle stage. Policies can be optional or required and can also validate that tasks associated with the stage are complete. These policies are automatically validated when a user requests stage approval or can be manually validated by users. The collective status of these policies gives a Compliance Status for an artifact which measures the current percentage of required policies which pass validation. The administrator assigns policies to lifecycle stages during the creation of lifecycle processes. For details about setting up and performing lifecycle validation, see "How to Define Policies" in the *Administration Guide* and "How to Review/Validate Policies" in the *User Guide*.

  Systinet includes a default Lifecycle Validation Task, scheduled to run once a day, which automatically validates all artifacts in governance against the policies that apply to their current lifecycle stage. For details about managing this task, see "Administration Task Management" in the *Administration Guide*.

- **Manual Validation**

  You can use the Policy Report feature of the Reports tab to perform ad-hoc manual validation. You can set up a policy report which validates a selected set of artifacts against a set of selected technical policies. For details, see "Policy Reports" on page 85.

- **Artifact Form Validation**

  The administrator can configure a set of technical policies that validate an artifact whenever it is created or modified. You can use this type of validation to ensure that new and amended artifacts contain a minimum set of descriptive data such as keywords and impact type, and enforce rules such as the versioning schema. Each artifact type can have its own set of form validation. For details about configuring form validation, see "How to Manage Artifact Form Validation" in the *Administration Guide*.

- **IDE Integration**

  HP Software also provides a set of IDE plugins with integrated policy validation functionality. For details, see *Plugin for Eclipse* and *Plugin for Visual Studio*.

The administrator of the top-level domain is responsible for policy management. For procedural details, see the following topics described in the *Administration Guide*:

- "How to Manage Technical Policies"
- "How to Manage Assertions"

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Concepts Guide (Systinet 10.01)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to docteam_systinet@hp.com.

We appreciate your feedback!