Functional Specification for the Object Manager (OM)

Resource Item

Version 2.0.4

Author HKA

Last changed 14.02.2007

Revision History

2.0.4	14.02.2007	HKA	All changes related to comments from MPGL included.
2.0.3	02.02.2007	HKA	Changes in chapter 2.
2.0.2	26.01.2007	HKA	Changes in "retrieveItems": filter name="item" renamed to filter name="objid".
2.0.1	18.01.2007	HKA	Changes in "General properties" and "create", "submit", "release" and "withdraw". (Done after first delivery of this document to ZIM.)
2.0	12.01.2007	HKA	Added Item resource chapter with interface description for Item resource. All changes integrated which were discussed and agreed in several meetings with ZIM and FIZ
1.9	03.11.2006	MSC	Created document. Added SOAP and REST overview and Object Manager chapter.

Table of Contents

List of Tables

Chapter 1. Overview

The eSciDOc Framework consists of resources. A resource is accessible by the REST Interface and by a ResourceHandler with SOAP. A resource which is a subset of an other resource is called subresource. A resource which is NO subresource is called base resource. With SOAP subresources can be accessed by the handler of their base resource. In principle it is possible to create, retrieve, update and delete a resource. Some but not all subresources allow all four operations.

A resource is represented as a XML document. There is a format description in XMLSchema for each base resource which includes the definitions of the subresources.

A retrieved representation of a (base or sub) resource reflects the current state of that resource. In order to change the state of a resource, the representation of that resource has to be changed and send back to the system in an **update** request.

In order to create a resource, build a representation of that resource as a XML document and send it to the system in a **create** request. A resource has a unique identifier. In a create request the unique identifier is not allowed. A unique identifier in a create request is not allowed and treated as an error. The response to a create request is the complete XML document containing the state of the created resource with a unique identifier assigned by the system, a status and a creation and last modification timestamp.

How to differentiate between retrieve, update, create and delete request and how to send the XML representation of a resource, see further for REST and SOAP. Both access methods use the same XML representations of resources as input and output.

1.1. REST

Each resource has a URI represented via XLink and is accessible with Hyper Text Transfer Protocol (HTTP). HTTP request methods are used to distinguish between create (PUT), update (PUT), retrieve (GET), and delete (DELETE). In order to create or update a resource the representation of that resource as a valid XML document has to be send in the request body. There is a restriction of the total size of that body.

In case of an error the answer contains a HTTP status code indicating the type of the error, a short description and a XML representation of the error. The XML representation of an error is described in XMLSchema [TODO document name].

Retrieve a resource. Send a HTTP GET request to the URI of the resource. The response to a HTTP GET request contains the HTTP status code 200 ("OK") and the XML representation of the resource in the response body.

Create or Update a resource. Send a HTTP PUT request to the URI of the resource. The body of the request must contain the XML representation of the resource and the value of the Content-Type header of the HTTP PUT request must be "text/xml". The HTTP status code of the response to a HTTP PUT request is 200 ("OK"). The response body contains the new or updated XML representation of the resource.

Delete a resource. Send a HTTP DELETE request to the URI of the resource. If successful the response to the HTTP DELETE request contains the HTTP status code 204 ("NO CONTENT") and no body.

Error handling. A REST request is submitted to eSciDoc by using the HTTP Protocol. If the execution of a request is errouneos an Exception is thrown. In this case the HTTP response status code is set to the http status code defined for this the exception, e.g. 404 if a resource

was not found by the framework. The text of the HTTP response will be an xml-representation of the exception. The possible status codes (combined with the causing exception) are listed in the documentaion of a REST method call. In the rare case the framework is not able to map a request to a specific resource method a MethodNotFoundException (404, "Method Not Found") is thrown and send back in the HTTP response. For further information on error and exception handling inside the framework please refer to the concept on exception handling [MHo06] .

1.2. SOAP

For each base resource exists a handler object which offers methods to create, retrieve, update and delete resources and their subresources. These methods - except the create methods - take a unique identifier to select a single resource.

A retrieve method returns the XML representation of a resource.

A create method takes the XML representation of a resource and returns the XML representation of the created resource.

An update method takes the XML representation of a resource and returns the XML representation of the updated resource.

Each handler object defines a service interface which is described in WSDL. For exception handling, method names and retrieving the service description see interface specification of the resources.

Exception handling. SOAP uses wsdl definitions of the services where it is specified how the request and the response of a service have to look like, including possible Exceptions. There are various tools on the market that can generate java-objects out of a wsdl (e.g. wsdl2java from the axis-framework). The response can automatically get filled into these java-objects by binding them to the specific responses. This enables an application to work with objects and not with xml. A java-application can now directly handle the eSciDoc-Exceptions by catching the generated exceptions. For further information on error and exception handling inside the framework please refer to the concept on exception handling [MHo06] and to the provided wsdl files.

1.3. Access to the Resource Handlers

All supported resources are accessible via REST and SOAP. All available methods to manipulate a certain resource via REST or SOAP are decribed in the document dedicated to this resource. Each resource offers its own SOAP service and has a wsdl interface description, e.g. the methods of the Item resource are accessible by the ItemHandlerService. The names of these services are listed in the description of the SOAP calls of the methods.

1.4. Resource and Task oriented Methods

According to the REST philosophy most methods of the framework are working on resources, e.g. Update an Item. The update method addresses a specific resource by its id and an updated XML representation of the specified Item. The XML representation includes all necessary information to perform the update, e.g. the timestamp of the latest modification required for optimistic locking.

In contrast to the resource oriented methods there are some exercises which are difficult to fulfill by performing a normal update. These methods are named task oriented methods. Possible tasks are the submission or the release of an Item or Container as well as the moving of obects from one Context to another or the filtered retrieval of objects. The complete list is found in the Section 4.2, “Task oriented Methods” . To be able to perform a task oriented meth-

od additional parameters are necessary. Examples are the filter criteria for a filtering retrieval method or the timestamp of the latest modification of the manipulated resource for optimistic locking purposes Section 2.3, “Optimistic Locking strategy” . These parameters are summarized in a special task-param XML structure. Examples are given in the documentation.

1.5. Working with Sub Resources

Every resource offers the facility to access its data by addressing sub resources. For example it is possible to retrieve and update a single metadata record of an Item instead of retrieving and updating the whole Item to correct a typo in the title.

To achieve the best performance in working with the eSciDoc framework and to minimize the arising overhead it is helpful to use the offered sub resources to retireve and update the required data.

1.6. Virtual Resources

In additon to the resources and related sub resources the eSciDoc framework supports "virtual" resources. These resources are not part of the stored objects but are created by the framework on request.

These resources are listed in the section "resources" of the XML-representations of each eSciDoc object.

Examples:

"/ir/container/container-id/toc/" is a sub resource of a container. In addition there exists a sub resource "/ir/container/container-id/resources/resource/toc-view" which is the "toc" enriched with additional data. The toc-view isn’t stored in the container but created when the sub resource is requested.

The framework eSciDoc may support other XML representations of objects e.x. METS. Such resources will also be listed in the section "resource".

Chapter 2. Object Manager (OM)

2.1. Overview of Object Manager (OM)

For the first release, the Object Manager (OM) offers services in the following areas:

- **Containers:** Containers offer the concept of aggregation, i.e. they can contain other (simple and complex) objects. Each container includes a structural map and can have a TOC and an AdminDescriptor. A container can be a collection or a bundle.
- **Items:** These are the objects in the eSciDoc system that comprises the institutional repository. An item is a simple or complex object like an article, a report or a translation.

  Items can have various statuses, which also have an influence on whether actions are applicable to them or not.

  In the first release, the Object Manager supports creation, retrieval, deletion, publishing and withdrawal of Items as well as various kinds of relations between Items.
- **Components:** Are used when an Item should not only store metadata about a specific content, but the content itself as digital objects. The Component will be handled as a separate part of the Item, enriched with metadata and licenses to facilitate handling and processing.

  In the first release, the Object Manager supports adding, retrieval, update and logically deletion of Components to / from Items.

  A Component can not exist without an Item.
- **Context:** These are objects that are used to administrate Items and Containers. They will not be versioned but can also have various statuses, which also has an influence on whether actions are applicable to them or not. Each context includes an AdminDescriptor.

  In the first release, the Object Manager supports the retrieval of Contexts as well as creating, updating, and deleting and furthermore adding, retrieving, and moving of Items and Containers to and from Contexts. New Contexts are created as "created".
- **Admin-Descriptor:** An AdminDescriptor is part of a Context or a Container. It keeps allowed and default settings for various parts of the Context or Container. It is stored as an object of its own even it is part to an Context or a Container.
- **Metadata:** A Metadata Record describes an Item or a Container in a given Metadata Schema. An Item or a Container must have at least the internal eSciDoc metadata record and may have more other Metadata Records in addition. A Metadata Record contains references to Metadata Elements of the respective Metadata Schema and values for these Metadata Elements. Depending on the Metadata Element definition, a Metadata Record may have more values for a single Metadata Element. Contexts do not have their own internal metadata record but one ore more default Metadata records for ingest and other purposes.

  Note: An Item or a Container has an internal Metadata Record as well as a set of properties. Both are versioned.

  In the first release, the Object Manager allows updating, and retrieving the eSciDoc internal Metadata record of Items and Containers.
- **properties:** In all methods except "create" the properties are read-only. In method "create"

some properties can be set (see methode create" for details).

The application may effect the values of some properties in using specific methods (like change "status" via method "release"), some properties are only changed by the framework (e.x. "last-modification-date", "creation-date").

- **Content Relations:** The Items stored in the Object Manager can be related to each other. The relations have types, names and are directed. Content Relations can have additional metadata and are versioned (e.g. the "is annotated by" between two Items.

  In the first release, Content Relations are not supported.

- **Structural Relations:** Items stored in the Object Manager belong always to a Context and may be part of a Container. The relations have types, names, and are directed. They have no metadata but are versioned (e.g. the "is administrated by" relation between Item and Context).

- **Persistent Identifiers:** Some of the object types in the eSciDoc framework will get assigned a persistent identifier (PID) at a certain point in their lifecycle, by which they can be identified no matter where they are currently located. These PIDs are also handled by the Object Manager.

  In the first release, the Object Manager supports creation and assignment of PIDs during the publishing process to Content Items as well as retrieval of Content Items via PID or mapping of PID to the internal ID.

- **Validation:** This means checking the validity of objects handled by the Object Manager.

- **Licenses:** are referenced from Components. They are stored in the Object Manager, to provide a list of all possible licenses.

  In the first release, Licenses are not supported.

2.2. Supported resources

For the first release, the Object Manager (OM) supports the following resources:

- Context
- Container
- Item

2.3. Optimistic Locking strategy

Updates on all objects managed by the Object Manager (OM) are executed using an *optimistic locking* strategy. Everyone may retrieve the contents and try to update the data afterwards. The update will only be successful if no one has changed the specified data in the meantime.

As a consequence it is necessary that the input data for the framework contains the timestamp of the object to update which is contained in the attribute "last-modification-date" of the root element of the retrieved object.

If only a subresource is updated the timestamp of the surrounding object has to be included as an attribute in the root element of the subresource.

Every update-method checks that the provided "last-modification-date" machtes the "last-

modification-date" currently saved in the system. If there is no match the update is rejected.

All delete-methods don't support the optimistic locking strategy.

2.4. General Information

2.4.1. Versioning (not supported)

The Object Manager will allow versioning of Items and Container but not in the first release. Although versioning is not part of this release this specification contains several information about versioning since it is important for the specification of the resource handling.

The concepts applied for the versioning of digital objects are explained in [MRFS06] . Only the latest version of an object can be updated.

2.4.2. License Types (not supported)

Even though License Types are an essential element of the EsciDoc framework object model their structure and content are not yet fully specified.

Nevertheless they are mentionend in the documentation of the methods where they are probably needed. But the associated functionality will not be implemenmted until the specification of the License Types is finished.

2.4.3. Content Types (not supported)

Even though Content Types are an essential element of the EsciDoc framework object model their structure and content are not yet fully specified. Nevertheless they are mentionend in the documentation of the methods where they are probably needed. But the associated functionality will not be implemenmted until the specification of the Content Types is finished.

Chapter 3. Resource Item

3.1. General Information

Updates of Items and Components.

In all methods XML-data is used as input and output. The related XML-schemas are located in www.escidoc.de/schemas/item/0.1/ [http://www.escidoc.de/schemas/item/0.1/] .

In the following sections only the filenames of the related schemas are specified.

These are the existing XML-schemas for item methods:

- item.xsd [http://www.escidoc.de/schemas/item/0.1/item.xsd]
- components.xsd [http://www.escidoc.de/schemas/item/0.1/components.xsd]
- md-records.xsd [http://www.escidoc.de/schemas/common/0.1/md-records.xsd]
- resources.xsd [http://www.escidoc.de/schemas/common/0.1/resources.xsd]
- xlink.xsd [http://www.escidoc.de/schemas/common/0.1/xlink.xsd]
- xml.xsd [http://www.escidoc.de/schemas/common/0.1/xml.xsd]

3.2. Description of Properties

3.2.1. Item

3.2.1.1. General Properties of an Item

Table 3.1. General Properties

Property	Description
objid	A unique identifier of the item within the system.
creation-date	This date is created by the framework when the framework stores the object the first time.
context	Every object has to be in exactly one "context". This is the link to that context.
content-type	Link to the "content type" object of that object.
creator	Link to the user who created the object, the framework adds this property.
lock-status	A user may lock an object. Valid status values are: • locked • unlocked
lock-owner	The user who set the lock-status to "locked".
lock-date	The date the lock-status was set to "locked".
withdrawal-comment	A comment which has to be given when an item is withdrawn.
current-version	Consists of the following elements: • number (The curent version number of the item)

Property	Description
	• date (The date of the current version of the item) • version-status (The status of the current version of the item) • valid-status
latest-version	Consists of the following elements: • number (The latest version number of the item) • date (The date of the latest version of the item)
latest-revision	Consists of the following elements: • number (The revision number of the item) • date (The date of the latest revision of the item) • pid
object-status	The state of the item. Valid status values are: • pending • submitted • released • withdrawn
pid	The persistent identifier of the released item. Handle Prefix + Item Id

3.2.1.2. Content-Type specific properties of an item

These properties are in the responsibility of the application.

They are maintained in section "content-type-specific" which is part of the section "properties".

Table 3.2. Properties for content type pubitem

Property	Description
ingestion-message	Contains messages arisen during ingestion.
automatic-submission-allowed	Defines whether automatic submission is allowed for the collection.Default value is true.

Table 3.3. Properties for content type SWB transcription and SWB translation

Property	Description
xml-valid	
nlp-source	

Table 3.4. Properties for content type SWB film and SWB image

Property	Description
allowed-download	

3.2.1.3. Additional Elements of an Item

Table 3.5. Elements of an Item

Element	Description
md-record	The content specific metadata of the object.
components	The components contain the binary data of that object and technical meta data.
relations	The content relations between this object and other objects.

3.2.2. Component

3.2.2.1. General Properties of a Component

Table 3.6. General Properties

Property	Description
objid	A unique identifier of the item within the system.
description	A description for the component.
creation-date	This date is created by the framework when the framework stores the component the first time.
latest-modification-date	This date is updated whenever the component is stored.
creator	Link to the user who created the object, the framework adds this property.
pid	The persistent identifier of the released binary data. Handle Prefix + Item Id.
visibility	The visibility for binary data of the item. Valid visibility values are: • Public • Private
component-type	The type of the component. Valid component types are: • abstract • pre-print • post-print • publisher version

Property	Description
	• copyright transfer agreement • correspondence • supplementary material
status	Status of the component.
file-name	Original filename of the binary content.
mime-type	Mimetype of the binary content.
file-size	Filesize of the binary content.

3.2.2.2. Additional Elements of a Component

Table 3.7. Elements of a Component

Element	Description
content	The binary data.

3.3. Virtual Resources

In additon to the resources and related sub resources the eSciDoc framework supports "virtual" resources. An item supports the following virtual resources:

- none

3.4. Download of binary content

While retrieving an item, the binary content is not provided inline in its representation. Instead, it can be downloaded by retrieving the "content" subresource of the item's components. This subresource is accessible via the REST interface, only. It is not possible to retrieve binary content via the SOAP interface.

The URI to retrieve a digital object of an item with id <item-id> that is stored in the item's component with id <component-id> is:

http://www.escidoc.de/ir/item/<item-id>/components/component/<component-id>/content

This link is provided within the respective component element in the item representation:

```
<item:item ...>
  <components:components ...>
    ...
    <components:component ...>
      ...
      <components:content xlink:type="simple" xlink:title="Content"
          xlink:href="/ir/item/<item-id>/components/<component-id>/content" />
      ...
    </components:component>
    ...
  </components:components>
</item:item>
```

3.5. Upload of binary content

The digital objects are stored while creating or updating an item. Therefore, the binary content has to be provided while calling this methods. There are two methods supported by the eSciDoc service:

- Providing a reference to the binary content
  - Reference to external file server
  - Reference to eSciDoc Staging Area
- Providing a binary content inline

Both methods may be combined while creating or updating an item.

3.5.1. Uploading by Reference

This method to upload binary content to the eSciDoc repository assumes that the files that shall be uploaded to eSciDoc are accessible from an external server. In this case, the files are referenced within the item representation in the xlink attributes of the empty content elements of the components:

```
<item:item ...>
  <components:components ...>
    ...
    <components:component ...>
      ...
      <components:content xlink:type="simple" xlink:title="Content"
          xlink:href="http://some.file.server/path/to/file"/>
      ...
    </components:component>
    ...
  </components:components>
</item:item>
```

While handling the creation of a new item or the update of an existing item, the eSciDoc base service fetches the binary contents from the provided URLs and stores them in the eSciDoc repository.

3.5.2. Uploading via Staging Area

This is a special kind of uploading by reference. As it can not be expected, that the files are always accessible from an external file server, eSciDoc provides a staging area that is accessible via the REST interface. To this staging area, the files can be uploaded by the services before calling the create or update methods:

```
HTTP PUT http://www.escidoc.de/st/staging-file
```

The result of this request is an XML structure describing the location of the uploaded file:

```
<staging-file:staging-file xlink:type="simple"
    xlink:href="http://www.escidoc.de/st/staging-file/<staging-file-id>">
```

In this case, the references to the files provided within the item representation are the URLs to the file in the staging area:

```
<item:item ...>
  <components:components ...>
    ...
    <components:component ...>
      ...
      <components:content xlink:type="simple" xlink:title="Content"
          xlink:href="http://www.escidoc.de/st/staging-file/<staging-file-id>"/>
      ...
    </components:component>
    ...
  </components:components>
</item:item>
```

3.5.3. Uploading Inline

The binary content that shall be stored for an item may be provided inline of the item's XML representation while creating or updating the item.

If this method is chosen, instead of specifying the xlink attributes, the binary content has to be encoded and provided within the "content" element of the component in that the content shall be stored:

```
<item:item ...>
  <components:components ...>
    ...
    <components:component ...>
      ...
      <components:content>
        Binary Content
      </components:content>
      ...
    </components:component>
    ...
  </components:components>
</item:item>
```

This uploading method is restricted: The provided data is only accepted if the overall size of the item representation is less than ESCIDOC_MAX_XML_SIZE . Otherwise, the create or update operation fails. The binary data has to be base64 encoded.

Chapter 4. Methods of Resource Item

4.1. Resource oriented Methods

4.1.1. create

Create an item

Prerequisites:

The provided XML data in the body is only accepted if the size is less than ESCIDOC_MAX_XML_SIZE.

Required input data:

- title of item (attribute "xlink:title" in root element)
- context (element of "properties")
- content type (element of "properties")
- content-type-specific (section of "properties")
- eSsciDoc Metadata Set (in "md-records:md-records")
- title of eSsciDoc Metadata Set (attribute "xlink:title" of "md-records:md-record")
- name of eSsciDoc Metadata Set (attribute "name" of "md-records:md-record")

Additional allowed input data:

- The section "components:components" with one or more sections "component".

For each section "component" the following data is required:

- title of component (attribute "xlink:title" in element "component")
- status (element of "component:properties")
- visibility (element of "component:properties")
- file-name (element of "component:properties")
- file-size (element of "component:properties")
- Binary content (element "content" of "components:component" with value of element "content" if binary content is "base64 encoded" delivered inline **or** attribute "xlink:href" with URL to binary content)

Additional allowed input data in component:

- mime-type (element of "component:properties")

No further elements, attributes or sections are allowed. If present they will cause an error.

Tasks:

- The XML data is validated against the XML-Schema of an item.
- It's checked wheather the contextid exists.
- Linked files are downloaded **or** extracted if inline delivered and the Components are created.
- The status of the item is set to "pending".
- The Version 1 of the item is created.
- item is added to the provided Context.

- The XML input data is updated and some new data is added (see below)
- The XML representation of the item corresponding to XML-schema is returned as output.

Updated and new data in XML representation:

- Content item (attributes "objid", "xlink:href" and "last-modification-date" in root element)
- Creation-date (element creation-date in properties)
- Status of the item (element status="pending" of properties)
- Creator (element "creator" of "properties" with attributes "xlink:href" and "xlink:title")
- Lock status (element lock-status="unlocked" of properties)
- Current version status (element version-status="pending" of "current-version")
- Current version number (element number="1" of "current-version")
- Current version date (element date='time stamp' of creation of version 1)
- Latest version status (element version-status="pending" of "latest-version")
- Latest version date (element date='time stamp' of latest modification in the system)
- Virtual resources (available virtual resources of item section "resources:resources")

For each section "components:component" the following data is updated and some new data is added :

- Content component (attributes "objid" and "xlink:href" of element "components:component")
- Creator (element "creator" of "components:properties" with attributes "xlink:href" and "xlink:title")
- Creation date (element "creation-date" of "components:properties")
- Last modification date (element "last-modification-date" of "components:properties")
- Referenz to binary content of the content component (attribute "href" of element "components:content")

Table 4.1. create via REST

HTTP Request	**PUT /ir/item**
Input from Uri	No input values
Input from Body	The XML representation of the item to be created corresponding to XML-schema "item.xsd".
Output	The XML representation of the created item corresponding to XML-schema "item.xsd".
Possible errors	451 content of component is missing (caused by MissingContentException) 404 Context Not Found (caused by ContextNotFoundException) 404 ContentType Not Found (caused by ContentTypeNotFoundException) 409 Readonly element is set by the user (caused by ReadonlyElementViolationException) 451 mandatory attribute value is missing (caused by MissingAttributeValueException) 450 schema validation failed (caused by XmlSchemaValidationException) 451 mandatory element text is missing (caused by MissingElementValueException)

HTTP Request	**PUT /ir/item**
	409 Readonly attribute is set by the user (caused by ReadonlyAttributeViolationException)
	302 Authentication failed.Redirect to login (caused by AuthenticationException)
	302 Authorization failed. Redirect to login (caused by AuthorizationException)
	450 Invalid XML (caused by InvalidXmlException)
	500 Internal Encoding Exception (caused by EncodingSystemException)
	500 Internal XML-Parser Error (caused by XmlParserSystemException)
	500 Internal File System Error (caused by FileSystemException)
	500 Internal Webserver Error (caused by WebserverSystemException)
	500 Internal Kowari Error (caused by KowariSystemException)
	500 Internal SQL-Database Error (caused by SqlDatabaseSystemException)
	500 Internal Fedora Error (caused by FedoraSystemException)
	451 Mandatory parameter is missing (caused by MissingParameterException)
	404 File Not Found (caused by FileNotFoundException)

Table 4.2. create via Soap

Method Signature	**String ItemHandlerService.create (String item)**
Parameter	*item*: The XML representation of the item to be created corresponding to XML-schema "item.xsd".
Output	The XML representation of the created item corresponding to XML-schema "item.xsd".
Possible errors	MissingContentException ContextNotFoundException ContentTypeNotFoundException ReadonlyElementViolationException MissingAttributeValueException XmlSchemaValidationException MissingElementValueException ReadonlyAttributeViolationException AuthenticationException AuthorizationException

Method Signature	**String ItemHandlerService.create (String item)**
	InvalidXmlException
	EncodingSystemException
	XmlParserSystemException
	FileSystemException
	WebserverSystemException
	KowariSystemException
	SqlDatabaseSystemException
	FedoraSystemException
	MissingParameterException
	FileNotFoundException

4.1.2. delete

Delete an item

Prerequisites:

The item has to be in object-status "pending" and must be unlocked, otherwise the removing of the item will fail.

Tasks:

- All content relations pointing from and to that item will be deleted.
- All components associated with that item will be deleted.
- The item will be deleted from IR.

Table 4.3. delete via REST

HTTP Request	**DELETE /ir/item/*<item-id>***
Input from Uri	*item-id*: The id of the item to be deleted.
Output	No return value
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 409 Resource already published (caused by AlreadyPublishedException) 409 Resource is locked (caused by LockingException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 450 Invalid status (caused by InvalidStatusException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Webserver Error (caused by WebserverSystemException) 500 Internal Encoding Exception (caused by EncodingSystemException) 500 Internal Kowari Error (caused by KowariSystemException) 500 Internal SQL-Database Error (caused by SqlDatabaseSystemException) 500 Internal Fedora Error (caused by FedoraSystemException) 404 Component Not Found (caused by ComponentNotFoundException) 451 Mandatory parameter is missing (caused by MissingParameterException)

Table 4.4. delete via Soap

Method Signature	**void ItemHandlerService.delete (String id)**
Parameter	*id*: The id of the item to be deleted.
Output	No return value
Possible errors	ItemNotFoundException AlreadyPublishedException LockingException AuthenticationException AuthorizationException InvalidStatusException XmlParserSystemException WebserverSystemException EncodingSystemException KowariSystemException SqlDatabaseSystemException FedoraSystemException ComponentNotFoundException MissingParameterException

4.1.3. retrieve

Retrieve an item

Prerequisites:

The item must exist

Tasks:

- The item is accessed using the provided reference.
- The XML representation to be returned for that item will not contain any binary content but references to them.
- The XML representation of the item corresponding to XML-schema is returned as output.

Table 4.5. retrieve via REST

HTTP Request	**GET /ir/item/*<item-id>***
Input from Uri	*item-id*: The id of the item to be retrieved.
Output	The XML representation of the retrieved item corresponding to XML-schema "item.xsd".
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 404 Component Not Found (caused by ComponentNotFoundException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Kowari Error (caused by KowariSystemException) 500 Internal Fedora Error (caused by FedoraSystemException) 451 Mandatory parameter is missing (caused by MissingParameterException) 500 Internal Encoding Exception (caused by EncodingSystemException)

Table 4.6. retrieve via Soap

Method Signature	**String ItemHandlerService.retrieve (String id)**
Parameter	*id*: The id of the item to be retrieved.
Output	The XML representation of the retrieved item corresponding to XML-schema "item.xsd".
Possible errors	ItemNotFoundException ComponentNotFoundException

Method Signature	**String ItemHandlerService.retrieve (String id)**
	AuthenticationException
	AuthorizationException
	XmlParserSystemException
	KowariSystemException
	FedoraSystemException
	MissingParameterException
	EncodingSystemException

4.1.4. update

Update an item

Prerequisites:

The provided XML data in the body is only accepted if the size is less than ESCIDOC_MAX_XML_SIZE.

The item must exist.

The the object-status is not "withdrawn".

The item is not locked.

Only the lastest version can be used here.

Required input data:

All sections, elements and attributes which are required in method "create" are required here, too.

The complete section "properties" has to be provieded as it was retrieved from IR and no values are allowed to be changed, removed or added. This section is read-only. Only elements of section "content-type-specific" (which is part of section "properties") and their values can be changed, deleted or added.

All other sections, elements and attributes which were part of the retrieved item must be present if they should remain in the new version. All parts which are not delivered in input are deleted.

Tasks:

- The XML data is validated against the XML-Schema of an item.
- Optimistic Locking criteria is checked.
- If new components are specified the linked files are downloaded **or** extracted if inline delivered this data is used and the new components are created.
- For existing components if new references are specified the linked files are downloaded **or** extracted if inline delivered this data is used and the components are updated.
- If changed, the internal metadata record is updated.
- A new version of the item is created.
- The XML input data is updated and some new data is added (see below)
- The XML representation of the item corresponding to XML-schema is returned as output.

Updated and new data in XML representation:

- Content item (attribute "last-modification-date" in root element)
- Virtual resources (available virtual resources of item section "resources:resources")

For each section "components:component" the following data is updated and some new data is added :

- Content component (if new: attributes "objid" and "xlink:href" of element "components:component")
- Creator (if new: element "creator" of "components:properties" with attributes "xlink:href" and

"xlink:title")
- Creation date (if new: element "creation-date" of "components:properties")
- Last modification date (element "last-modification-date" of "components:properties")
- Referenz to binary content of the content component (if new or changed: attribute "href" of element "components:content")

Note: The status of an item can't bet changed by this method nor can't it be moved to a different context. Please use the methods dedicated for this purpose.

Table 4.7. update via REST

HTTP Request	**PUT /ir/item/*<item-id>***
Input from Uri	*item-id*: The id of the item to be updated.
Input from Body	The XML representation of the item to be updated corresponding to XML-schema "item.xsd".
Output	The XML representation of the updated item corresponding to XML-schema "item.xsd".
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 404 File Not Found (caused by FileNotFoundException) 450 Invalid context (caused by InvalidContextException) 450 Invalid status (caused by InvalidStatusException) 409 Resource is locked (caused by LockingException) 450 schema validation failed (caused by XmlSchemaValidationException) 409 Resource is not published yet (caused by NotPublishedException) 451 license is missing (caused by MissingLicenceException) 404 Component Not Found (caused by ComponentNotFoundException) 451 content of component is missing (caused by MissingContentException) 409 Readonly element is set by the user (caused by ReadonlyElementViolationException) 451 mandatory attribute value is missing (caused by MissingAttributeValueException) 409 Resource already published (caused by AlreadyPublishedException) 409 Readonly attribute is set by the user (caused by ReadonlyAttributeViolationException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 450 Invalid XML (caused by InvalidXmlException)

HTTP Request	**PUT /ir/item/*<item-id>***
	500 Internal XML-Parser Error (caused by XmlParserSystemException)
	500 Internal Encoding Exception (caused by EncodingSystemException)
	500 Internal Integrity Error (caused by IntegritySystemException)
	500 Internal File System Error (caused by FileSystemException)
	500 Internal Webserver Error (caused by WebserverSystemException)
	500 Internal Kowari Error (caused by KowariSystemException)
	500 Internal SQL-Database Error (caused by SqlDatabaseSystemException)
	500 Internal Fedora Error (caused by FedoraSystemException)
	451 Mandatory parameter is missing (caused by MissingParameterException)

Table 4.8. update via Soap

Method Signature	**String ItemHandlerService.update (String id, String item)**
Parameter	*id*: The id of the item to be updated. *item*: The XML representation of the item to be updated corresponding to XML-schema "item.xsd".
Output	The XML representation of the updated item corresponding to XML-schema "item.xsd".
Possible errors	ItemNotFoundException FileNotFoundException InvalidContextException InvalidStatusException LockingException XmlSchemaValidationException NotPublishedException MissingLicenceException ComponentNotFoundException MissingContentException ReadonlyElementViolationException MissingAttributeValueException AlreadyPublishedException ReadonlyAttributeViolationException

Method Signature	**String ItemHandlerService.update (String id, String item)**
	AuthenticationException
	AuthorizationException
	InvalidXmlException
	XmlParserSystemException
	EncodingSystemException
	IntegritySystemException
	FileSystemException
	WebserverSystemException
	KowariSystemException
	SqlDatabaseSystemException
	FedoraSystemException
	MissingParameterException

4.2. Task oriented Methods

4.2.1. retrieveItems

Retrieves a list of items applying filters.

Prerequisites:

The items must exist

At least one filter containing a value must be specified.

Tasks:

- Check whether all filter names are valid.
- The items are accessed using the provided filters.
- The XML representation to be returned for all item will not contain any binary content but references to them.
- The XML representation of the list of all items corresponding to XML-schema is returned as output.

Table 4.9. retrieveItems via REST

HTTP Request	**POST /ir/items/filter**
Input from Uri	No input values
Input from Body	The filter criteria to select the items corresponding to "filter.xsd". Valid filter criteria names are: "items" "creator" "related (TRUE | FALSE)" "status". an example: <param> <filter name="items"> <id>escidoc:23232</id> <id>escidoc:12121</id> </filter> <filter name="creator">escidoc:14141</filter> <filter name="related">true</filter> <filter name="status">submitted</filter> </param> If multiple filters are specified they are linked using logical AND.
Output	The XML representation of the created list of items corresponding to XML-schema "item-list.xsd".
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 404 Component Not Found (caused by ComponentNotFoundException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 451 Mandatory parameter is missing (caused by MissingParameterException)

HTTP Request	POST /ir/items/filter
	500 Internal Kowari Error (caused by KowariSystemException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Fedora Error (caused by FedoraSystemException) 450 Invalid XML (caused by InvalidXmlException) 500 Internal Encoding Exception (caused by EncodingSystemException)

Table 4.10. retrieveItems via Soap

Method Signature	String ItemHandlerService.retrieveItems (String taskParam)
Parameter	*taskParam*: The filter criteria to select the items corresponding to "filter.xsd". Valid filter criteria names are: "items" "creator" "related (TRUE | FALSE)" "status". an example: <param> <filter name="items"> <id>escidoc:23232</id> <id>escidoc:12121</id> </filter> <filter name="creator">escidoc:14141</filter> <filter name="related">true</filter> <filter name="status">submitted</filter> </param> If multiple filters are specified they are linked using logical AND.
Output	The XML representation of the created list of items corresponding to XML-schema "item-list.xsd".
Possible errors	ItemNotFoundException ComponentNotFoundException AuthenticationException AuthorizationException MissingParameterException KowariSystemException XmlParserSystemException FedoraSystemException InvalidXmlException EncodingSystemException

4.2.2. retrieveItemRefs

Retrieves a list of item references applying filters.

Prerequisites:

The items must exist

At least one filter containing a value must be specified.

Tasks:

- Check whether all filter names are valid.
- The items are accessed using the provided filters.
- The XML representations of the list of all item references corresponding to XML-schema is returned as output.

Table 4.11. retrieveItemRefs via REST

HTTP Request	**POST /ir/items/filter/refs**
Input from Uri	No input values
Input from Body	The filter criteria to select the items corresponding to "filter.xsd". Valide filter criteria names are: "items" "creator" "related (TRUE | FALSE)" "status". an example: <param> <filter name="objid"> <id>escidoc:23232</id> <id>escidoc:12121</id> </filter> <filter name="creator">escidoc:14141</filter> <filter name="related">true</filter> <filter name="status">submitted</filter> </param> If multiple filters are specified they are link using logical AND.
Output	The XML representation of the created list of references to items corresponding to XML-schema "item-refs-list.xsd".
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 404 Component Not Found (caused by ComponentNotFoundException) 451 Mandatory parameter is missing (caused by MissingParameterException) 500 Internal Kowari Error (caused by KowariSystemException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 450 Invalid XML (caused by InvalidXmlException)

Table 4.12. retrieveItemRefs via Soap

Method Signature	**String ItemHandlerService.retrieveItemRefs (String taskParam)**
Parameter	*taskParam*: The filter criteria to select the items corresponding to "filter.xsd". Valide filter criteria names are: "items" "creator" "related (TRUE | FALSE)" "status". an example: <param> <filter name="objid"> <id>escidoc:23232</id> <id>escidoc:12121</id> </filter> <filter name="creator">escidoc:14141</filter> <filter name="related">true</filter> <filter name="status">submitted</filter> </param> If multiple filters are specified they are link using logical AND.
Output	The XML representation of the created list of references to items corresponding to XML-schema "item-refs-list.xsd".
Possible errors	ItemNotFoundException ComponentNotFoundException MissingParameterException KowariSystemException XmlParserSystemException InvalidXmlException

4.2.3. submit

Submit an item

Prerequisites:

The item must exist.

The the latest-version-status is "pending".

The item is not locked.

The object-status is not "withdrawn".

Only the lastest version can be used here.

Tasks:

- Optimistic Locking criteria is checked.
- The status of the latest-version is changed to "submitted".
- Latest version date is updated.
- No new version is created.
- No data is returned.

Table 4.13. submit via REST

HTTP Request	**POST /ir/item/<*item-id*>/submit**
Input from Uri	*item-id*: The id of the item to be submitted.
Input from Body	The timestamp of the last modification of the item in a XML-Structure: `<param last-modification-date="1967-08-13T12:00:00.000+01:00" />`
Output	No return value
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 409 Resource is locked (caused by LockingException) 450 Invalid status (caused by InvalidStatusException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Encoding Exception (caused by EncodingSystemException) 500 Internal Webserver Error (caused by WebserverSystemException) 500 Internal Kowari Error (caused by KowariSystemException) 500 Internal SQL-Database Error (caused by SqlDatabaseSystemException)

HTTP Request	**POST /ir/item/<*item-id*>/submit**
	500 Internal Fedora Error (caused by FedoraSystemException) 451 Mandatory parameter is missing (caused by MissingParameterException)

Table 4.14. submit via Soap

Method Signature	**void ItemHandlerService.submit (String id, String taskParam)**
Parameter	*id*: The id of the item to be submitted. *taskParam*: The timestamp of the last modification of the item in a XML-Structure: `<param last-modification-date="1967-08-13T12:00:00.000+01:00" />`
Output	No return value
Possible errors	ItemNotFoundException LockingException InvalidStatusException AuthenticationException AuthorizationException XmlParserSystemException EncodingSystemException WebserverSystemException KowariSystemException SqlDatabaseSystemException FedoraSystemException MissingParameterException

4.2.4. release

Release an item

Prerequisites:

The item must exist.

The latest-version-status is "submitted".

The item is not locked.

The object-status is not "withdrawn".

Only the lastest version can be used here.

Tasks:

- Optimistic Locking criteria is checked.
- The status of the item is changed to "released".
- No new version is created.
- No data is returned.

Table 4.15. release via REST

HTTP Request	**POST /ir/item/<*item-id*>/release**
Input from Uri	*item-id*: The id of the item to be released.
Input from Body	The timestamp of the last modification of the item. `<param last-modification-date="1967-08-13T12:00:00.000+01:00" />`
Output	No return value
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 409 Resource is locked (caused by LockingException) 450 Invalid status (caused by InvalidStatusException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Encoding Exception (caused by EncodingSystemException) 500 Internal Application Server Error (caused by ApplicationServerSystemException) 500 Internal Webserver Error (caused by WebserverSystemException) 500 Internal Kowari Error (caused by KowariSystemException)

HTTP Request	**POST /ir/item/<*item-id*>/release**
	500 Internal SQL-Database Error (caused by SqlDatabaseSystemException) 500 Internal Fedora Error (caused by FedoraSystemException) 451 Mandatory parameter is missing (caused by MissingParameterException)

Table 4.16. release via Soap

Method Signature	**void ItemHandlerService.release (String id, String taskParam)**
Parameter	*id*: The id of the item to be released. *taskParam*: The timestamp of the last modification of the item. `<param last-modification-date="1967-08-13T12:00:00.000+01:00" />`
Output	No return value
Possible errors	ItemNotFoundException LockingException InvalidStatusException AuthenticationException AuthorizationException XmlParserSystemException EncodingSystemException ApplicationServerSystemException WebserverSystemException KowariSystemException SqlDatabaseSystemException FedoraSystemException MissingParameterException

4.2.5. withdraw

Withdraw an item

Prerequisites:

The item must exist.

The the object-status is "released".

The item is not locked.

The object-status is not "withdrawn".

Only the lastest version can be used here.

Tasks:

- Optimistic Locking criteria is checked.
- The status of the item is changed to "withdrawn".
- No new version is created.
- No data is returned.

Table 4.17. withdraw via REST

HTTP Request	**POST /ir/item/<*item-id*>/withdraw**
Input from Uri	*item-id*: The id of the item to be withdrawn.

Input from Body	The timestamp of the last modification of the item.
	
	`<param last-modification-date="1967-08-13T12:00:00.000+01:00">`
	`  <withdraw-comment>Withdraw comment.</withdraw-comment>`
	`</param>`

Output	No return value
Possible errors	404 Item Not Found (caused by ItemNotFoundException) 409 Resource is not published yet (caused by NotPublishedException) 409 Resource is locked (caused by LockingException) 409 Resource already withdrawn (caused by AlreadyWithdrawnException) 302 Authentication failed.Redirect to login (caused by AuthenticationException) 302 Authorization failed. Redirect to login (caused by AuthorizationException) 450 Invalid status (caused by InvalidStatusException) 500 Internal XML-Parser Error (caused by XmlParserSystemException) 500 Internal Encoding Exception (caused by EncodingSystemException) 500 Internal Application Server Error (caused by ApplicationServerSystemExcep-

HTTP Request	POST /ir/item/<*item-id*>/withdraw
	tion) 500 Internal Webserver Error (caused by WebserverSystemException) 500 Internal Kowari Error (caused by KowariSystemException) 500 Internal SQL-Database Error (caused by SqlDatabaseSystemException) 500 Internal Fedora Error (caused by FedoraSystemException) 451 Mandatory parameter is missing (caused by MissingParameterException)

Table 4.18. withdraw via Soap

Method Signature	void ItemHandlerService.withdraw (String id, String taskParam)

Parameter	*id*: The id of the item to be withdrawn.
	taskParam: The timestamp of the last modification of the item.
	`<param last-modification-date="1967-08-13T12:00:00.000+01:00">`
	`  <withdraw-comment>Withdraw comment.</withdraw-comment>`
	`</param>`

Output	No return value
Possible errors	ItemNotFoundException NotPublishedException LockingException AlreadyWithdrawnException AuthenticationException AuthorizationException InvalidStatusException XmlParserSystemException EncodingSystemException ApplicationServerSystemException WebserverSystemException KowariSystemException SqlDatabaseSystemException FedoraSystemException MissingParameterException

References

[MHo06] Michael Hoppe. *Exceptions in eSciDoc*. 2006.

[MRFS06] Matthias Razum. Frank Schwichtenberg. *Concept for Versioning and Object Identification*. 2006.

[TTe06] Torsten Tetteroo. *Concept for User Management*. 2006.

[TTe06-1] Torsten Tetteroo. *Concept for Authentication and Authorization*. 2006.