|  |  |
| --- | --- |
| Title | CLARIN Federated Content Search (CLARIN-FCS) – Core Specification |
| Version | 1.0 |
| Authors | Oliver Schonefeld (IDS, CLARIN-D) (Editor) Thomas Eckart (ASV Leipzig, CLARIN-D) Thomas Kisler (BAS, CLARIN-D) Christoph Draxler (BAS, CLARIN-D) Kai Zimmer (BBAW, CLARIN-D,) Matej Ďurčo (CLARIN Center Vienna, CLARIN-AT) Yana Panchenko (Tübingen University, CLARIN-D) Hanna Hedeland (HZSK, CLARIN-D) Andre Blessing (Stuttgart University, CLARIN-D) Olha Shkaravska (MPI for Psycholinguistics, CLARIN-NL) |
| Date | 2014-04-07 |
| Status | Draft for approval by SCCTC |
| Distribution | SCCTC |
| ID | CE-2014-0316 |

# Introduction

The goal of the CLARIN Federated Content Search (CLARIN-FCS) - Core specification is to introduce an interface specification that decouples the search engine functionality from its exploitation, i.e. user-interfaces, third-party applications, and to allow services to access heterogeneous search engines in a uniform way.

## Terminology

The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in [RFC2119].

## Glossary

Aggregator

A module or service to dispatch queries to repositories and collect results.

CLARIN-FCS, FCS

CLARIN federated content search, an interface specification to allow searching within resource content of repositories.

Client

A software component, which implements the interface specification to query Endpoints, i.e. an aggregator or a user-interface.

CQL

Contextual Query Language, previously known as Common Query Language, is a formal language for representing queries to information retrieval systems such as search engines, bibliographic catalogs and museum collection information.

Data View

A Data View is a mechanism to support different representations of search results, e.g. a "hits with highlights" view, an image or a geolocation.

Data View Payload, Payload

The actual content encoded within a Data View, i.e. a CMDI metadata record or a KML encoded geolocation.

Endpoint

A software component, which implements the CLARIN-FCS interface specification and translates between CLARIN-FCS and a search engine.

Hit

A piece of data returned by a Search Engine that matches the search criterion. What is considered a Hit highly depends on Search Engine.

Interface Specification

Common harmonized interface and suite of protocols that repositories need to implement.

PID

A persistent identifier is a long-lasting reference to a digital object.

Repository

A software component at a CLARIN center that stores resources (= data) and information about these resources (= metadata).

Repository Registry

A separate service that allows registering Repositories and their Endpoints and provides information about these to other components, e.g. an Aggregator. The CLARIN Center Registry[[1]](#footnote-1) is an implementation of such a repository registry.

Resource

A searchable and addressable entity at an Endpoint, such as a text corpus or a multi-modal corpus.

Resource Fragment

A smaller unit in a Resource, i.e. a sentence in a text corpus or a time interval in an audio transcription.

Result Set

An (ordered) set of hits that match a search criterion produced by a search engine as the result of processing a query.

Search Engine

A software component within a repository, that allows for searching within the repository contents.

SRU

Search and Retrieve via URL, is a protocol for Internet search queries. Originally introduced by Library of Congress [LOC-SRU12], later standardization process moved to OASIS [OASIS-SRU12].

## Normative References

RFC2119

Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119, March 1997,  
<http://www.ietf.org/rfc/rfc2119.txt>

XML-Namespaces

Namespaces in XML 1.0 (Third Edition), W3C, 8 December 2009,  
<http://www.w3.org/TR/2009/REC-xml-names-20091208/>

OASIS-SRU-Overview

searchRetrieve: Part 0. Overview Version 1.0, OASIS, January 2013,  
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part0-overview/searchRetrieve-v1.0-os-part0-overview.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part0-overview/searchRetrieve-v1.0-os-part0-overview.html)), ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part0-overview/searchRetrieve-v1.0-os-part0-overview.pdf))

OASIS-SRU-APD

searchRetrieve: Part 1. Abstract Protocol Definition Version 1.0, OASIS, January 2013,  
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part1-apd/searchRetrieve-v1.0-os-part1-apd.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part1-apd/searchRetrieve-v1.0-os-part1-apd.html)) ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part1-apd/searchRetrieve-v1.0-os-part1-apd.pdf))

OASIS-SRU12

searchRetrieve: Part 2. SRU searchRetrieve Operation: APD Binding for SRU 1.2 Version 1.0, OASIS, January 2013,  
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part2-sru1.2/searchRetrieve-v1.0-os-part2-sru1.2.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part2-sru1.2/searchRetrieve-v1.0-os-part2-sru1.2.html)) ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part2-sru1.2/searchRetrieve-v1.0-os-part2-sru1.2.pdf))

OASIS-CQL

searchRetrieve: Part 5. CQL: The Contextual Query Language version 1.0, OASIS, January 2013,   
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part5-cql/searchRetrieve-v1.0-os-part5-cql.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part5-cql/searchRetrieve-v1.0-os-part5-cql.html)) ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part5-cql/searchRetrieve-v1.0-os-part5-cql.pdf))

SRU-Explain

searchRetrieve: Part 7. SRU Explain Operation version 1.0, OASIS, January 2013,  
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part7-explain/searchRetrieve-v1.0-os-part7-explain.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part7-explain/searchRetrieve-v1.0-os-part7-explain.html)) ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part7-explain/searchRetrieve-v1.0-os-part7-explain.pdf))

SRU-Scan

searchRetrieve: Part 6. SRU Scan Operation version 1.0, OASIS, January 2013,   
<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part6-scan/searchRetrieve-v1.0-os-part6-scan.doc> ([HTML](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part6-scan/searchRetrieve-v1.0-os-part6-scan.html)) ([PDF](http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part6-scan/searchRetrieve-v1.0-os-part6-scan.PDF))

LOC-SRU12

SRU Version 1.2: SRU Search/Retrieve Operation, Library of Congress,   
<http://www.loc.gov/standards/sru/sru-1-2.html>

LOC-DIAG

SRU Version 1.2: SRU Diagnostics List, Library of Congress,  
<http://www.loc.gov/standards/sru/diagnostics/diagnosticsList.html>

CLARIN-FCS-DataViews

CLARIN Federated Content Search (CLARIN-FCS) - Data Views 1.0, SCCTC FCS Task-Force, March 2014,   
<http://www.clarin.eu/fcs/add/link/here>

## Non-Normative References

RFC6838

Media Type Specifications and Registration Procedures, IETF RFC 6838, January 2013,   
<http://www.ietf.org/rfc/rfc6838.txt>

RFC3023

XML Media Types, IETF RFC 3023, January 2001,   
<http://www.ietf.org/rfc/rfc3023.txt>

## Typographic and XML Namespace conventions

The following typographic conventions for XML fragments will be used throughout this specification:

* <prefix:Element>  
  An XML element with the Generic Identifier Element that is bound to an XML namespace denoted by the prefix prefix.
* @attr  
  An XML attribute with the name attr
* string  
  The literal string must be used either as element content or attribute value.

Endpoints and Clients MUST adhere to the [XML-Namespaces] specification. The CLARIN-FCS interface specification generally does not dictate whether XML elements should be serialized in their prefixed or non-prefixed syntax, but Endpoints MUST ensure that the correct XML namespace is used for elements and that XML namespaces are declared correctly. Clients MUST be agnostic regarding syntax for serializing the XML elements, i.e. if the prefixed or un-prefixed variant was used, and SHOULD operate solely on expanded names, i.e. pairs of namespace name and local name.

The following XML namespace names and prefixes are used throughout this specification. The column "Recommended Syntax" indicates which syntax variant SHOULD be used by the Endpoint to serialize the XML response.

|  |  |  |  |
| --- | --- | --- | --- |
| **Prefix** | **Namespace Name** | **Comment** | **Recommended Syntax** |
| fsc | http://clarin.eu/fcs/resource | CLARIN-FCS Resources | prefixed |
| ed | http://clarin.eu/fcs/endpoint-description | CLARIN-FCS Endpoint Description | prefixed |
| hits | http://clarin.eu/fcs/dataview/hits | CLARIN-FCS Generic Hits | prefixed |
| sru | http://www.loc.gov/zing/srw/ | SRU | prefixed |
| diag | http://www.loc.gov/zing/srw/diagnostic/ | SRU Diagnostics | prefixed |
| zr | http://explain.z3950.org/dtd/2.0/ | SRU/ZeeRex Explain | prefixed |

# CLARIN-FCS Interface Specification

The CLARIN-FCS Interface Specification defines a set of capabilities, an extensible result format and a set of required operations. CLARIN-FCS is built on the SRU/CQL standard and additional functionality required for CLARIN-FCS is added through SRU/CQL's extension mechanisms.

Specifically, the CLARIN-FCS Interface Specification consists of two components, a set of formats and a transport protocol. The Endpoint component is a software component that acts as a bridge between the formats that are sent by a Client using the Transport Protocol, and a Search Engine. The Search Engine is a custom software component that allows the search of language resources in a Repository. The Endpoint basically implements the transport protocol and acts as a mediator between the CLARIN-FCS specific formats and the idiosyncrasies of Search Engines of the individual Repositories. The following figure illustrates the overall architecture:

+---------+

| Client |

+---------+

/|\

|

-------------------------

| SRU / CQL |

| w/CLARIN-FCS extensions |

-------------------------

|

\|/

+-----------------------------------------+

| | Endpoint /|\ |

| | | |

| --------------- ------------------ |

| | Translate CQL | | Translate Result | |

| --------------- ------------------ |

| | | |

| \|/ | |

+-----------------------------------------+

/|\

|

\|/

+---------------------------+

| Search Engine |

+---------------------------+

In general, the work flow in CLARIN-FCS is as follows: a Client submits a query to an Endpoint. The Endpoint translates the query from CQL to the query dialect used by the Search Engine and submits the translated query to the Search Engine. The Search Engine processes the query and generates a result set, i.e. it compiles a set of hits that match the search criterion. The Endpoint then translates the results from the Search Engine-specific result set format to the CLARIN-FCS result format and sends it to the Client.

The following sections describe the CLARIN-FCS capabilities, the query and result formats, how SRU/CQL is used as a transport protocol in the context of CLARIN-FCS and the required CLARIN-FCS specific extensions to SRU.

## Capabilities

Because CLARIN-FCS aims to integrate several heterogeneous Search Engines it needs to support the different features of the Search Engines, e.g. some only allow simple search while other support wild-cards or regular expressions. Also, the available resources have different properties, e.g. some resources provide Part-Of-Speech annotations, while others are transcripts of an audio signal or lexicographic resources consisting of entries. To accommodate for these different features, CLARIN-FCS defines several Capabilities. A Capability defines a certain feature that is part of CLARIN-FCS, e.g. what kind of queries are supported. Each Endpoint implements some (or all) of these Capabilities. The Endpoint will announce the capabilities it provides to allow a Client to auto-tune itself (see section 2.3). Each Capability is identified by a Capability Identifier, which uses the URI syntax.

The following Capabilities are defined by CLARIN-FCS:

Basic Search (Capability Identifier: http://clarin.eu/fcs/capability/basic-search)

Endpoints MUST support term-only queries.   
Endpoints SHOULD support terms combined with boolean operator queries (AND and OR), including sub-queries. Endpoints MAY also support NOT or PROX operator queries. If the Endpoint does not support a query, i.e. the used operators are not supported by the Endpoint, it MUST return an appropriate error message using the appropriate SRU diagnostic (see [LOC-DIAG]).   
Examples for valid CQL queries for the Basic Search:

cat

"cat"

cat AND dog

"grumpy cat"

"grumpy cat" AND dog

"grumpy cat" OR "lazy dog"

cat AND (mouse OR "lazy dog")

The Endpoint MUST perform the query on an annotation tier that, which makes the most sense for the user, i.e. the textual content for a text corpus resource or the orthographic transcription of a spoken language corpus. Endpoints SHOULD perform the query case-sensitive.   
**NOTE**: In CQL, a term can be a single token or a phrase, i.e. tokens separated by spaces. If a term contains spaces, it needs to be quoted.   
**NOTE**: CLARIN-FCS requires Endpoints to be able to parse all of CQL and, if they don't support a certain CQL feature, to generate the appropriate error message (see section 3.1). Especially, if an Endpoint only supports Basic Search, it MUST NOT silently accept queries that include CQL features besides term-only and terms combined with boolean operator queries, i.e. queries involving context sets, etc.

Endpoints MUST implement the Basic Search Capability.

Endpoints MUST NOT invent custom Capability Identifiers and MUST only use the values defined above.

**NOTE**: The current CLARIN-FCS specification only defines the Basic Search Capability. Future versions of the CLARIN-FCS specification will support more sophisticated queries such as selecting annotation tiers, expanding of tags, or mapping of data categories and thus define more Capabilities. A future CLARIN-FCS specification my also introduce the term "Profile" as a simple way to refer to a certain sub-set of Capabilities.

## Result Format

The Search Engine will produce a result set containing several hits as the outcome of processing a query. The Endpoint MUST serialize these hits in the CLARIN-FCS result format. Endpoints are REQUIRED to adhere to the principle, that one hit MUST be serialized as one CLARIN-FCS result record and MUST NOT combine several hits in one CLARIN-FCS result record. E.g., if a query matches five different sentences within one text (= the resource), the Endpoint must serialize them as five SRU records each with one Hit each referencing the same containing Resource (see section 3.4).

CLARIN-FCS uses a customized format for returning results. Resource and Resource Fragments serve as containers for hit results, which are presented in one or more Data View. The following section describes the Resource format and Data View format and section 3.4 will describe, how hits are embedded within SRU responses.

### Resource and ResourceFragment

To encode search results, CLARIN-FCS supports two building blocks:

Resources

A Resource is a searchable and addressable entity at the Endpoint, such as a text corpus or a multi-modal corpus. A resource SHOULD be a self-contained unit, i.e. not a single sentence in a text corpus or a time interval in an audio transcription, but rather a complete document from a text corpus or a complete audio transcription.

Resource Fragments

A Resource Fragment is a smaller unit in a Resource, i.e. a sentence in a text corpus or a time interval in an audio transcription.

A Resource SHOULD be the most precise unit of data that is directly addressable as a "whole". A Resource SHOULD contain a Resource Fragment, if the hit consists of just a part of the Resource unit (for example if the hit is a sentence within a large text). A Resource Fragment SHOULD be addressable within a resource, i.e. it has an offset or a resource-internal identifier. Using Resource Fragments is OPTIONAL, but Endpoints are encouraged to use them. If the Endpoint encodes a hit with a Resource Fragment, the actual hit SHOULD be encoded as a Data View that within the Resource Fragment.

Endpoints SHOULD always provide a link to the resource itself, i.e. each Resource or Resource Fragment SHOULD be identified by a persistent identifier or providing a URI, that is unique for Endpoint. Even if direct linking is not possible, i.e. due to licensing issues, the Endpoints SHOULD provide a URI to link to a web-page describing the corpus or collection, including instruction on how to obtain it. Endpoints SHOULD provide links that are as specific as possible (and logical), i.e. if a sentence within a resource cannot be addressed directly, the Resource Fragment SHOULD NOT contain a persistent identifier or a URI.

If the Endpoint can provide both, a persistent identifier as well as a URI, for either Resource or Resource Fragment, they SHOULD provide both. When working with results, Clients SHOULD prefer persistent identifiers over regular URIs.

Resource and Resource Fragment are serialized in XML and Endpoints MUST generate responses that are valid according to the XML schema "[Resource.xsd](https://trac.clarin.eu/browser/FederatedSearch/schema/Resource.xsd)" ([download](https://trac.clarin.eu/browser/FederatedSearch/schema/Resource.xsd?format=txt)). A Resource is encoded in the form of a <fcs:Resource> element, a Resource Fragment in the form of a <fcs:ResourceFragment> element. The content of a Data View is wrapped in a <fcs:DataView> element. <fcs:Resource> is the top-level element and MAY contain zero or more <fcs:DataView> elements and MAY contain zero or more <fcs:ResourceFragment> elements. A <fcs:ResourceFragment> element MUST contain one or more <fcs:DataView> elements.

The elements <fcs:Resource>, <fcs:ResourceFragment> and <fcs:DataView> MAY carry a @pid and/or a @ref attribute, which allows linking to the original data represented by the Resource, Resource Fragment, or Data View. A @pid attribute MUST contain a valid persistent identifier, a @ref MUST contain valid URI, i.e. a "plain" URI without the additional semantics of being a persistent reference. If the Endpoint cannot provide a @pid attribute for a <fcs:Resource>, they SHOULD provide a @ref attribute. Endpoint SHOULD add either a @pid or @ref attribute to either the <fcs:Resource> or the <fcs:ResourceFragment> element, if possible to both elements. Endpoints are RECOMMENDED to give @pid attributes, if they can provide them.

Endpoints MUST use the identifier http://clarin.eu/fcs/resource for the responseItemType (= content for the <sru:recordSchema> element) in SRU responses.

Endpoints MAY serialize hits as multiple Data Views, however they MUST provide the Generic Hits (HITS) Data View either encoded as a Resource Fragment (if applicable), or otherwise within the Resource (if there is no reasonable Resource Fragment). Other Data Views SHOULD be put in a place that is logical for their content (as is to be determined by the Endpoint), e.g. a metadata Data View would most likely be put directly below Resource and a Data View representing some annotation layers directly around the hit is more likely to belong within a Resource Fragment.

Example 1 shows a simple hit, which is encoded in one Data View of type Generic Hits embedded within a Resource. The type of the Data View is identified by the MIME type application/x-clarin-fcs-hits+xml. The Resource is referenceable by the persistent identifier http://hdl.handle.net/4711/08-15.

Example 1: A simple hit

<fcs:Resource xmlns:fcs="http://clarin.eu/fcs/resource"

pid="http://hdl.handle.net/4711/00-15">

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<!-- data view payload omitted -->

</fcs:DataView>

</fcs:Resource>

Example 2 shows a hit encoded as a Resource Fragment embedded within a Resource. The actual hit is again encoded as one Data View of type Generic Hits. The hit is not directly referenceable, but the Resource, in which hit occurred, is referenceable by the persistent identifier http://hdl.handle.net/4711/08-15. In contrast to Example 1, the Endpoint decided to provide a "semantically richer" encoding and embedded the hit using a Resource Fragment within the Resource to indicate that the hit is a part of a larger resource, e.g. a sentence in a text document.

Example 2: A hit encoded as Resource Fragment within a Resource

<fcs:Resource xmlns:fcs="http://clarin.eu/fcs/resource"

pid="http://hdl.handle.net/4711/08-15">

<fcs:ResourceFragment>

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<!-- data view payload omitted -->

</fcs:DataView>

</fcs:ResourceFragment>

</fcs:Resource>

The most complex Example 3 is similar to Example 2, i.e. it shows a hit is encoded as one Generic Hits Data View in a Resource Fragment, which is embedded in a Resource. In contrast to Example 2, another Data View of type CMDI is embedded directly within the Resource. The Endpoint can use this type of Data View to directly provide CMDI metadata about the Resource to Clients. All entities of the Hit can be referenced by a persistent identifier and a URI. The complete Resource is referenceable by either the persistent identifier http://hdl.handle.net/4711/08-15 or the URI http://repos.example.org/file/text\_08\_15.html and the CMDI metadata record in the CMDI Data View is referenceable either by the persistent identifier http://hdl.handle.net/4711/08-15-1 or the URI http://repos.example.org/file/08\_15\_1.cmdi. The actual hit in the Resource Fragment is also directly referenceable by either the persistent identifier http://hdl.handle.net/4711/00-15-2 or the URI http://repos.example.org/file/text\_08\_15.html#sentence2.

Example 3: A complex hit

<fcs:Resource xmlns:fcs="http://clarin.eu/fcs/resource"

pid="http://hdl.handle.net/4711/08-15"

ref="http://repos.example.org/file/text\_08\_15.html">

<fcs:DataView type="application/x-cmdi+xml"

pid="http://hdl.handle.net/4711/08-15-1"

ref="http://repos.example.org/file/08\_15\_1.cmdi">

<!-- data view payload omitted -->

</fcs:DataView>

<fcs:ResourceFragment pid="http://hdl.handle.net/4711/08-15-2"

ref="http://repos.example.org/file/text\_08\_15.html#sentence2">

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<!-- data view payload omitted -->

</fcs:DataView>

</fcs:ResourceFragment>

</fcs:Resource>

### Data View

A Data View serves as a container for encoding the actual search results (the data fragments relevant to search) within CLARIN-FCS. Data Views are designed to allow for different representations of results, i.e. they are deliberately kept open to allow further extensions with more supported Data View formats. This specification only defines a most basic Data View for representing search results, called Generic Hits (see below). More Data Views are defined in the supplementary specification [CLARIN-FCS-DataViews].

The content of a Data View is called Payload. Each Payload is typed and the type of the Payload is recorded in the @type attribute of the <fcs:DataView> element. The Payload type is identified by a MIME type ([RFC6838], [RFC3023]). If no existing MIME type can be used, implementers SHOULD define a proper private mime type.

The Payload of a Data View can either be deposited inline or by reference. In the case of inline, it MUST be serialized as an XML fragment below the <fcs:DataView> element. This is the preferred method for payloads that can easily be serialized in XML. Deposition by reference is meant for content that cannot easily be deposited inline, i.e. binary content (like images). In this case, the Data View MUST include a @ref or @pid attribute that links location for Clients to download the payload. This location SHOULD be openly accessible, i.e. data can be downloaded freely without any need to perform a login.

Data Views are classified into a send-by-default and a need-to-request delivery policy. In case of the send-by-default delivery policy, the Endpoint MUST send the Data View automatically, i.e. Endpoints MUST unconditionally include the Data View when they serialize a response to a search request. In the case of need-to-request, the Client must explicitly request the Endpoint to include this Data View in the response. This enables the Endpoint to not generate and serialize Data Views that are "expensive" in terms of computational power or bandwidth for every response. To request such a Data View, a Client MUST submit a comma separated list of Data View identifiers (see section 2.3) in the x-fcs-dataviews extra request parameter with the searchRetrieve request. If a Client requests a Data View that is not valid for the search context, the Endpoint MUST generate a non-fatal diagnostic http://clarin.eu/fcs/diagnostic/4 ("Requested Data View not valid for this resource"). The details field of the diagnostic MUST contain the MIME type of the Data View that was not valid. If more than one requested Data View is invalid, the Endpoint MUST generate a separate non-fatal diagnostic http://clarin.eu/fcs/diagnostic/4 for each of the requested Data Views.

The description of every Data View contains a recommendation as to how the Endpoint should handle the payload delivery, i.e. if a Data View is by default considered send-by-default or need-to-request. Endpoint MAY choose to implement different policy. The relevant information which policy is implemented by an Endpoint for a specific Data View is part of the Endpoint Description (see section 2.3). For each Data View, a Recommended Short Identifier is defined, that Endpoint SHOULD use for an identifier of the Data View in the list of supported Data Views in the Endpoint Description

The Generic Hits Data View is mandatory, thus all Endpoints MUST implement this it and provide search results represented in the Generic Hits Data View. Endpoints MUST implement the Generic Hits Data View with the send-by-default delivery policy.

**NOTE**: The examples in the following sections show only the payload with the enclosing <fcs:DataView> element of a Data View. Of course, the Data View must be embedded either in a <fcs:Resource> or a <fcs:ResourceFragment> element. The @pid and @ref attributes have been omitted for all inline payload types.

#### Generic Hits (HITS)

|  |  |
| --- | --- |
| **Description** | The representation of the hit |
| **MIME type** | application/x-clarin-fcs-hits+xml |
| **Payload Disposition** | inline |
| **Payload Delivery** | send-by-default (REQUIRED) |
| **Recommended Short Identifier** | hits (RECOMMENDED) |

The Generic Hits Data View serves as the most basic agreement in CLARIN-FCS for serialization of search results and MUST be implemented by all Endpoints. In many cases, this Data View can only serve as an (lossy) approximation, because resources at Endpoints are very heterogeneous. For instance, the Generic Hits Data View is probably not the best representation for a hit result in a corpus of spoken language, but an architecture like CLARIN-FCS requires one common representation to be implemented by all Endpoints, therefore this Data View was defined. The Generic Hits Data View supports multiple markers for supplying highlighting for an individual hit, e.g. if a query contains a (boolean) conjunction, the Endpoint can use multiple markers to provide individual highlights for the matching terms. An Endpoint MUST NOT use this Data View to aggregate several hits within one resource. Each hit SHOULD be presented within the context of a complete sentence. If that is not possible due to the nature of the type of the resource, the Endpoint MUST provide an equivalent reasonable unit of context (e.g. within a phrase of an orthographic transcription of an utterance). The <hits:Hit> element within the <hits:Result> element is not enforced by the XML schema, but Endpoints are RECOMMENDED to use it. The XML fragment of the Generic Hits payload MUST be valid according to the XML schema "[DataView-Hits.xsd](https://trac.clarin.eu/browser/FederatedSearch/schema/DataView-Hits.xsd)" ([download](https://trac.clarin.eu/browser/FederatedSearch/schema/DataView-Hits.xsd?format=txt)).

<!-- potential @pid and @ref attributes omitted -->

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<hits:Result xmlns:hits="http://clarin.eu/fcs/dataview/hits">

The quick brown <hits:Hit>fox</hits:Hit> jumps over the lazy dog.

</hits:Result>

</fcs:DataView>

Example 4: Generic Data View with single hit marker

Example 5: Generic Hits Data View with multiple hit markers

<!-- potential @pid and @ref attributes omitted -->

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<hits:Result xmlns:hits="http://clarin.eu/fcs/dataview/hits">

The quick brown <hits:Hit>fox</hits:Hit> jumps over the↩

lazy <hits:Hit>dog</hits:Hit>.

</hits:Result>

</fcs:DataView>

## Endpoint Description

Endpoints need to provide information about their capabilities to support auto-configuration of Clients. The Endpoint Description mechanism provides the necessary facility to provide this information to the Clients. Endpoints MUST encode their capabilities using an XML format and embed this information into the SRU/CQL protocol as described in section 3.2. The XML fragment generated by the Endpoint for the Endpoint Description MUST be valid according to the XML schema "[Endpoint-Description.xsd](https://trac.clarin.eu/browser/FederatedSearch/schema/Endpoint-Description.xsd)" ([download](https://trac.clarin.eu/browser/FederatedSearch/schema/Endpoint-Description.xsd?format=txt)).

The XML fragment for Endpoint Description is encoded as an <ed:EndpointDescription> element, that contains the following attributes and children:

* one @version attribute (REQUIRED) on the <ed:EndpointDescription> element. The value of the @version attribute MUST be 1.
* one <ed:Capabilities> element (REQUIRED) that contains one or more <ed:Capability> elements.  
  The content of the <ed:Capability> element is a Capability Identifier, that indicates the capabilities, that are supported by the Endpoint. For valid values for the Capability Identifier, see section 2.1. This list MUST NOT include duplicate values.
* one <ed:SupportedDataViews> (REQUIRED)   
  A list of Data Views that are supported by this Endpoint. This list is composed of one or more <ed:SupportedDataView> elements. The content of a <ed:SupportedDataView> MUST be the MIME type of a supported Data View, e.g. application/x-clarin-fcs-hits+xml. Each <ed:SupportedDataView> element MUST carry a @id and a @delivery-policy attribute. The value of the @id attribute is later used in the <ed:Resource> element to indicate, which Data View is supported by a resource (see below). Endpoints SHOULD use the recommended short identifier for the Data View. The @delivery-policy indicates, the Endpoint's delivery policy, for that Data View. Valid values are send-by-default for the send-by-default and need-to-request for the need-to-request delivery policy.   
  This list MUST NOT include duplicate entries, i.e. no MIME type must appear more than once. The value of the @id attribute MUST NOT contain the characters , (comma) or ; (semicolon)
* one <ed:Resources> element (REQUIRED)   
  A list of (top-level) resources that are available, i.e. searchable, at the Endpoint. The <ed:Resources> element contains one or more <ed:Resource> elements (see below). The Endpoint MUST declare at least one (top-level) resource.
* The <ed:Resource> element contains a basic description of a resource that is available at the Endpoint. A resource is a searchable entity, e.g. a single corpus. The <ed:Resources> has a mandatory @pid attribute that contains persistent identifier of the resource. This value MUST be the same as the MdSelfLink of the CMDI record describing the resource. The <ed:Resources> element contains the following children:
* one or more <ed:Title> elements (REQUIRED)   
  A human readable title for the resource. A REQUIRED @xml:lang attribute indicates the language of the title. An English version of the title is REQUIRED. The list of titles MUST NOT contain duplicate entries for the same language.
* zero or more <ed:Description> elements (OPTIONAL)   
  An optional human-readable description of the resource. It SHOULD be at most one sentence. A REQUIRED @xml:lang attribute indicates the language of the description. If supplied, an English version of the description is REQUIRED. The list of descriptions MUST NOT contain duplicate entries for the same language.
* zero or one <ed:LandingPageURI> element (OPTIONAL)   
  A link to a website for the resource, e.g. a landing page for a resource, i.e. a web-site that describes a corpus.
* one <ed:Languages> element (REQUIRED)   
  The (relevant) languages available within the resource. The <ed:Languages> element contains one or more <ed:Language> elements. The content of a <ed:Language> element MUST be a ISO 639-3 three letter language code. This element should be repeated for all languages (relevant) available within the resource, however this list MUST NOT contain duplicate entries.
* one <ed:AvailableDataViews> element (REQUIRED)   
  The Data Views that are available for the resource. The <ed:AvailableDataViews> MUST carry a @ref attribute, that contains a whitespace separated list of id values, that correspond to value of @id attribute on <ed:SupportedDataView> elements.   
  In case of sub-resources, each Resource SHOULD support all Data Views that are supported by the parent resource. However, every resource MUST declare all available Data Views independently, i.e. there is no implicit inheritance semantic.
* zero or one <ed:Resources> element (OPTIONAL)   
  If a resource has searchable sub-resources, the Endpoint MUST supply additional finer grained resource elements, which are wrapped in a <ed:Resources> element. A sub-resource is a searchable entity within a resource, e.g. a sub-corpus.

Example 6 shows a simple Endpoint Description for an Endpoint that only supports the Basic Search Capability and only provides the Generic Hits Data View, which is indicated by a <ed:SupportedDataView> element. This element carries a @id attribute with a value of hits, the recommended value for the short identifier, and indicates a delivery policy of send-by-default by the @delivery-policy attribute. It only provides one top-level resource identified by the persistent identifier http://hdl.handle.net/4711/0815. The resource has a title as well as a description in German and English. A landing page is located at http://repos.example.org/corpus1.html. The predominant language in the resource contents is German. Only the Generic Hits Data View is supported for this resource, because the <ed:AvailableDataViews> element only references the <ed:SupporedDataView> element with the @id with a value of hits.

Example 6: A simple Endpoint Description

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

<ed:EndpointDescription xmlns:ed=http://clarin.eu/fcs/endpoint-description  
 version="1">

<ed:Capabilities>

<ed:Capability>↩  
 http://clarin.eu/fcs/capability/basic-search</ed:Capability>

</ed:Capabilities>

<ed:SupportedDataViews>

<ed:SupportedDataView id="hits"

delivery-policy="send-by-default">  
 application/x-clarin-fcs-hits+xml</ed:SupportedDataView>

</ed:SupportedDataViews>

<ed:Resources>

<!-- just one top-level resource at the Endpoint -->

<ed:Resource pid="http://hdl.handle.net/4711/0815">

<ed:Title xml:lang="de">Goethe Corpus</ed:Title>

<ed:Title xml:lang="en">Goethe Korpus</ed:Title>

<ed:Description xml:lang="de">Der Goethe Korpus des↩

IDS Mannheim</ed:Description>

<ed:Description xml:lang="en">The Goethe corpus of↩  
 IDS Mannheim</ed:Description>

<ed:LandingPageURI>↩  
 http://repos.example.org/corpus1.html</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits"/>

</ed:Resource>

</ed:Resources>

</ed:EndpointDescription>

The more complex Example 7 show an Endpoint Description for an Endpoint that, similar to Example 6, supports the Basic Search capability. In addition to the Generic Hits Data View, it also supports the CMDI Data View. The delivery polices are send-by-default for the Generic Hits Data View and need-to-request for the CMDI Data View. The Endpoint has two top-level resources (identified by the persistent identifiers http://hdl.handle.net/4711/0815 and http://hdl.handle.net/4711/0816. The second top-level resource has two independently searchable sub-resources, identified by the persistent identifier http://hdl.handle.net/4711/0816-1 and http://hdl.handle.net/4711/0816-2. All resources are described using several properties, like title, description, etc. The first top-level resource provides only the Generic Hits Data View, while the other top-level resource including its children provide the Generic Hits and the CMDI Data Views.

Example 7: A more complex Endpoint Description

<ed:EndpointDescription xmlns:ed="http://clarin.eu/fcs/endpoint-description" version="1">

<ed:Capabilities>

<ed:Capability>http://clarin.eu/fcs/capability/basic-search</ed:Capability>

</ed:Capabilities>

<ed:SupportedDataViews>

<ed:SupportedDataView id="hits" delivery-policy="send-by-default">↩  
 application/x-clarin-fcs-hits+xml</ed:SupportedDataView>

<ed:SupportedDataView id="cmdi" delivery-policy="need-to-request">↩  
 application/x-cmdi+xml</ed:SupportedDataView>

</ed:SupportedDataViews>

<ed:Resources>

<!-- top-level resource 1 -->

<ed:Resource pid="http://hdl.handle.net/4711/0815">

<ed:Title xml:lang="de">Goethe Corpus</ed:Title>

<ed:Title xml:lang="en">Goethe Korpus</ed:Title>

<ed:Description xml:lang="de">Der Goethe Korpus des↩  
 IDS Mannheim</ed:Description>

<ed:Description xml:lang="en">The Goethe corpus of↩  
 IDS Mannheim</ed:Description>

<ed:LandingPageURI>↩  
 http://repos.example.org/corpus1.html</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits"/>

</ed:Resource>

<!-- top-level resource 2 -->

<ed:Resource pid="http://hdl.handle.net/4711/0816">

<ed:Title xml:lang="de">↩  
 Mannheimer Morgen newspaper Corpus</ed:Title>

<ed:Title xml:lang="en">↩  
 Zeitungskorpus des Mannheimer Morgen</ed:Title>

<ed:LandingPageURI↩>  
 http://repos.example.org/corpus2.html</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits cmdi"/>

<ed:Resources>

<!-- sub-resource 1 of top-level resource 2 -->

<ed:Resource pid="http://hdl.handle.net/4711/0816-1">

<ed:Title xml:lang="de">↩  
 Mannheimer Morgen newspaper Corpus (before 1990)</ed:Title>

<ed:Title xml:lang="en">↩  
 Zeitungskorpus des Mannheimer Morgen (vor 1990)</ed:Title>

<ed:LandingPageURI>↩  
 http://repos.example.org/corpus2.html#sub1</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits cmdi"/>

</ed:Resource>

<!-- sub-resource 2 of top-level resource 2 -->

<ed:Resource pid="http://hdl.handle.net/4711/0816-2">

<ed:Title xml:lang="de">↩  
 Mannheimer Morgen newspaper Corpus (after 1990)</ed:Title>

<ed:Title xml:lang="en">↩  
 Zeitungskorpus des Mannheimer Morgen (nach 1990)</ed:Title>

<ed:LandingPageURI>↩  
 http://repos.example.org/corpus2.html#sub2</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits cmdi"/>

</ed:Resource>

</ed:Resources>

</ed:Resource>

</ed:Resources>

</ed:EndpointDescription>

## Endpoint Custom Extensions

Endpoints can add custom extensions, i.e. custom data, to the Result Format. This extension mechanism can for example be used to provide hints for an (XSLT/XQuery) application that works directly on CLARIN-FCS, e.g. to allow it to generate back and forward links to navigate in a result set.

An Endpoint MAY add arbitrary XML fragments to the extension hooks provided in the <fcs:Resource> element (see the XML schema for "Resource.xsd"). The XML fragment for the extension MUST use a custom XML namespace name for the extension. Endpoints MUST NOT use XML namespace names that start with the prefixes http://clarin.eu, http://www.clarin.eu/, https://clarin.eu or http://www.clarin.eu/.

A Client MUST ignore any custom extensions it does not understand and skip over these XML fragments when parsing the Endpoint's response.

The appendix 5.3 contains an example, how an extension could be implemented.

# CLARIN-FCS to SRU/CQL binding

## SRU/CQL

SRU (Search/Retrieve via URL) specifies a general communication protocol for searching and retrieving records and the CQL (Contextual Query Language) specifies an extensible query language. CLARIN-FCS is built on SRU 1.2. A subsequent specification may be built on SRU 2.0.

Endpoints and Clients MUST implement the SRU/CQL protocol suite as defined in [OASIS-SRU-Overview], [OASIS-SRU-APD], [OASIS-CQL], [SRU-Explain], [SRU-Scan], especially with respect to:

* Data Model,
* Query Model,
* Processing Model,
* Result Set Model, and
* Diagnostics Model

Endpoints and Clients MUST implement the APD Binding for SRU 1.2, as defined in [OASIS-SRU-12]. Endpoints and Clients MAY also implement APD binding for version 1.1 or version 2.0.

Endpoints and Clients MUST use the following XML namespace names (namespace URIs) for serializing responses:

* http://www.loc.gov/zing/srw/ for SRU response documents, and
* http://www.loc.gov/zing/srw/diagnostic/ for diagnostics within SRU response documents.

CLARIN-FCS deviates from the OASIS specification [OASIS-SRU-Overview] and [OASIS-SRU-12] to ensure backwards comparability with SRU 1.2 services as they were defined by the [LOC-SRU12].

Endpoints or Clients MUST support CQL conformance Level 2 (as defined in [OASIS-CQL, section 6]), i.e. be able to parse (Endpoints) or serialize (Clients) all of CQL and respond with appropriate error messages to the search/retrieve protocol interface.

**NOTE**: this does not imply, that Endpoints are required to support all of CQL, but rather that they are able to parse all of CQL and generate the appropriate error message, if a query includes a feature they do not support.

Endpoints MUST generate diagnostics according to [OASIS-SRU-12, Appendix C] for error conditions or to indicate unsupported features. Unfortunately, the OASIS specification does not provides a comprehensive list of diagnostics for CQL-related errors. Therefore, Endpoints MUST use diagnostics from [LOC-DIAG, section "Diagnostics Relating to CQL"] for CQL related errors.

Endpoints MUST support the HTTP GET [OASIS-SRU-12, Appendix B.1] and HTTP POST [OASIS-SRU-12, Appendix B.2] lower level protocol binding. Endpoints MAY also support the [SOAP OASIS-SRU-12], Appendix B.3 binding.

## Operation *explain*

The explain operation of the SRU protocol serves to announce server capabilities and to allow clients to configure themselves automatically. This operation is used similarly.

The Endpoint MUST respond to a explain request by a proper explain response. As per [SRU-Explain], the response MUST contain one <sru:record> element that contains an SRU Explain record. The <sru:recordSchema> element MUST contain the literal http://explain.z3950.org/dtd/2.0/, i.e. the official identifier for Explain records.

According to the Capabilities supported by the Endpoint the Explain record MUST contain the following elements:

Basic-Search Capability

* <zr:serverInfo> as defined in [SRU-Explain] (REQUIRED)
* <zr:databaseInfo> as defined in [SRU-Explain] (REQUIRED)
* <zr:schemaInfo> as defined in [SRU-Explain] (REQUIRED). This element MUST contain an element <zr:schema> with an @identifier attribute with a value of http://clarin.eu/fcs/resource and an @name attribute with a value of fcs.
* <zr:configInfo> is OPTIONAL
* Other capabilities may define how the <zr:indexInfo> element is to be used, therefore it is NOT RECOMMENDED for Endpoints to use it in custom extensions.

To support auto-configuration in CLARIN-FCS, the Endpoint MUST provide support Endpoint Description. The Endpoint Description is included in explain response utilizing SRUs extension mechanism, i.e. by embedding an XML fragment into the <sru:extraResponseData> element. The Endpoint MUST include the Endpoint Description only if the Client performs an explain request with the extra request parameter x-fcs-endpoint-description with a value of true. If the Client performs an explain request without supplying this extra request parameter the Endpoint MUST NOT include the Endpoint Description. The format of the Endpoint Description XML fragment is defined in section 2.3.

The following example shows a request and response to an explain request with added extra request parameter x-fcs-endpoint-description:

HTTP GET request: Client → Endpoint:

http://repos.example.org/fcs-endpoint?operation=explain&↩  
 version=1.2&x-fcs-endpoint-description=true

HTTP Response: Endpoint → Client:

<?xml version='1.0' encoding='utf-8'?>

<sru:explainResponse xmlns:sru="http://www.loc.gov/zing/srw/">

<sru:version>1.2</sru:version>

<sru:record>

<sru:recordSchema>http://explain.z3950.org/dtd/2.0/</sru:recordSchema>

<sru:recordPacking>xml</sru:recordPacking>

<sru:recordData>

<zr:explain xmlns:zr="http://explain.z3950.org/dtd/2.0/">

<!-- <zr:serverInfo > is REQUIRED -->

<zr:serverInfo protocol="SRU" version="1.2" transport="http">

<zr:host>repos.example.org</zr:host>

<zr:port>80</zr:port>

<zr:database>fcs-endpoint</zr:database>

</zr:serverInfo>

<!-- <zr:databaseInfo> is REQUIRED -->

<zr:databaseInfo>

<zr:title lang="de">Goethe Corpus</zr:title>

<zr:title lang="en" primary="true">Goethe Korpus</zr:title>

<zr:description lang="de">Der Goethe Korpus des  
 IDS Mannheim.</zr:description>

<zr:description lang="en" primary="true">The Goethe corpus of  
 IDS Mannheim.</zr:description>

</zr:databaseInfo>

<!-- <zr:schemaInfo> is REQUIRED -->

<zr:schemaInfo>

<zr:schema identifier="http://clarin.eu/fcs/resource" name="fcs">

<zr:title lang="en" primary="true">CLARIN Federated  
 Content Search</zr:title>

</zr:schema>

</zr:schemaInfo>

<!-- <zr:configInfo> is OPTIONAL -->

<zr:configInfo>

<zr:default type="numberOfRecords">250</zr:default>

<zr:setting type="maximumRecords">1000</zr:setting>

</zr:configInfo>

</zr:explain>

</sru:recordData>

</sru:record>

<!-- <sru:echoedExplainRequest> is OPTIONAL -->

<sru:echoedExplainRequest>

<sru:version>1.2</sru:version>

<sru:baseUrl>http://repos.example.org/fcs-endpoint</sru:baseUrl>

</sru:echoedExplainRequest>

<sru:extraResponseData>

<ed:EndpointDescription  
 xmlns:ed="http://clarin.eu/fcs/endpoint-description" version="1">

<ed:Capabilities>

<ed:Capability>  
 http://clarin.eu/fcs/capability/basic-search</ed:Capability>

</ed:Capabilities>

<ed:SupportedDataViews>

<ed:SupportedDataView id="hits"  
 delivery-policy="send-by-default">  
 application/x-clarin-fcs-hits+xml</ed:SupportedDataView>

</ed:SupportedDataViews>

<ed:Resources>

<!-- just one top-level resource at the Endpoint -->

<ed:Resource pid="http://hdl.handle.net/4711/0815">

<ed:Title xml:lang="de">Goethe Corpus</ed:Title>

<ed:Title xml:lang="en">Goethe Korpus</ed:Title>

<ed:Description xml:lang="de">Der Goethe Korpus des  
 IDS Mannheim.</ed:Description>

<ed:Description xml:lang="en">The Goethe corpus of  
 IDS Mannheim.</ed:Description>

<ed:LandingPageURI>  
 http://repos.example.org/corpus1.html</ed:LandingPageURI>

<ed:Languages>

<ed:Language>deu</ed:Language>

</ed:Languages>

<ed:AvailableDataViews ref="hits"/>

</ed:Resource>

</ed:Resources>

</ed:EndpointDescription>

</sru:extraResponseData>

</sru:explainResponse>

## Operation *scan*

The scan operation of the SRU protocol is currently not used in the Basic Search capability of CLARIN-FCS. Future capabilities may use this operation, therefore it NOT RECOMMENDED for Endpoints to define custom extensions that use this operation.

## Operation *searchRetrieve*

The searchRetrieve operation of the SRU protocol is used for searching in the Resources that are provided by the Endpoint. The SRU protocol defines the serialization of request and response formats in [OASIS-SRU-12]. In SRU, search result hits are encoded down to a record level, i.e. the <sru:record> element, and SRU allows records to be serialized in various formats, so called record schemas.

Endpoints MUST support the CLARIN-FCS record schema (see section 2.2) and MUST use the value http://clarin.eu/fcs/resource for the responseItemType ("record schema identifier"). Endpoints MUST represent exactly one hit within the Resource as one SRU record, i.e. <sru:record> element.

The following example shows a request and response to a searchRetrieve request with a term-only query for "cat":

HTTP GET request: Client → Endpoint:

http://repos.example.org/fcs-endpoint?operation=searchRetrieve&↩  
 version=1.2&query=cat

HTTP Response: Endpoint → Client:

<?xml version='1.0' encoding='utf-8'?>

<sru:searchRetrieveResponse xmlns:sru="http://www.loc.gov/zing/srw/">

<sru:version>1.2</sru:version>

<sru:numberOfRecords>6</sru:numberOfRecords>

<sru:records>

<sru:record>

<sru:recordSchema>http://clarin.eu/fcs/resource</sru:recordSchema>

<sru:recordPacking>xml</sru:recordPacking>

<sru:recordData>

<fcs:Resource xmlns:fcs=http://clarin.eu/fcs/resource  
 pid="http://hdl.handle.net/4711/08-15">

<fcs:ResourceFragment>

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<hits:Result xmlns:hits="http://clarin.eu/fcs/dataview/hits">

The quick brown <hits:Hit>cat</hits:Hit> jumps over the  
 lazy dog.

</hits:Result>

</fcs:DataView>

</fcs:ResourceFragment>

</fcs:Resource>

</sru:recordData>

<sru:recordPosition>1</sru:recordPosition>

</sru:record>

<!-- more <sru:records> omitted for brevity -->

</sru:records>

<!-- <sru:echoedSearchRetrieveRequest> is OPTIONAL -->

<sru:echoedSearchRetrieveRequest>

<sru:version>1.2</sru:version>

<sru:query>cat</sru:query>

<sru:xQuery xmlns="http://www.loc.gov/zing/cql/xcql/">

<searchClause>

<index>cql.serverChoice</index>

<relation>

<value>=</value>

</relation>

<term>cat</term>

</searchClause>

</sru:xQuery>

<sru:startRecord>1</sru:startRecord>

<sru:baseUrl>http://repos.example.org/fcs-endpoint</sru:baseUrl>

</sru:echoedSearchRetrieveRequest>

</sru:searchRetrieveResponse>

In general, the Endpoint is REQUIRED to accept an unrestricted search and SHOULD perform the search operation on all Resources that are available at the Endpoint. If that is for some reason not feasible, e.g. performing an unrestricted search would allocate too many resources, the Endpoint MAY independently restrict the search to a scope that it can handle. If it does so, it MUST issue a non-fatal diagnostics http://clarin.eu/fcs/diagnostic/2 ("Resource set too large. Query context automatically adjusted."). The details field of diagnostics MUST contain the persistent identifier of the resources to which the query scope was limited to. If the Endpoint limits the query scope to more than one resource, it MUST generate a separate non-fatal diagnostic http://clarin.eu/fcs/diagnostic/2 for each of the resources.

The Client can request the Endpoint to restrict the search to a sub-resource of these Resources. In this case, the Client MUST pass a comma-separated list of persistent identifiers in the x-fcs-context extra request parameter of the searchRetrieve request. The Endpoint MUST then restrict the search to those Resources, which are identified by the persistent identifiers passed by the Client. If a Client requests too many resources for the Endpoint to handle with x-fcs-context, the Endpoint MAY issue a fatal diagnostic http://clarin.eu/fcs/diagnostic/3 ("Resource set too large. Cannot perform Query.") and terminate processing. Alternatively, the Endpoint MAY also automatically adjust the scope and issue a non-fatal diagnostic http://clarin.eu/fcs/diagnostic/2 (see above). And Endpoint MUST NOT issue a http://clarin.eu/fcs/diagnostic/3 diagnostic in response to a request, if a Client performed the request without the x-fcs-context extra request parameter.

The Client can extract all valid persistent identifiers from the @pid attribute of the <ed:Resource> element, obtained by the explain request (see section 3.2 and section 2.3). The list of persistent identifiers can get extensive, but a Client can use the HTTP POST method instead of HTTP GET method for submitting the request.

For example, to restrict the search to the Resource with the persistent identifier http://hdl.handle.net/4711/0815 the Client must issue the following request:

http://repos.example.org/fcs-endpoint?operation=searchRetrieve&↩  
 version=1.2&query=cat&x-fcs-context=http://hdl.handle.net/4711/0815

To restrict the search to the Resources with the persistent identifier http://hdl.handle.net/4711/0815 and http://hdl.handle.net/4711/0816-2 the Client must issue the following request:

http://repos.example.org/fcs-endpoint?operation=searchRetrieve&↩  
 version=1.2&query=cat&x-fcs-context=http://hdl.handle.net/4711/0815,↩  
 http://hdl.handle.net/4711/0816-2

If an invalid persistent identifier is passed by the Client, the Endpoint MUST issue a http://clarin.eu/fcs/diagnostic/1 diagnostic, i.e. add the appropriate XML fragment to the <sru:diagnostics> element of the response. The Endpoint MAY treat this condition as fatal, i.e. just issue the diagnostic and perform no search, or it MAY treat it as non-fatal and perform the search.

If a Client wants to request one or more Data Views, that are handled by Endpoint with the need-to-request delivery policy, it MUST pass a comma-separated list of Data View identifier in the x-fcs-dataviews extra request parameter of the *searchRetrieve* request. A Client can extract valid values for the Data View identifiers from the @id attribute of the <ed:SupportedDataView> elements in the Endpoint Description of the Endpoint (see section 3.2 and section 2.3).

For example, to request the CMDI Data View from an Endpoint that has an Endpoint Description, as described in Example 7, a Client would need to use the Data View identifier cmdi and submit the following request:

http://repos.example.org/fcs-endpoint?operation=searchRetrieve&↩  
 version=1.2&query=cat&x-fcs-dataviews=cmdi

If an invalid Data View identifier is passed by the Client, the Endpoint MUST issue a http://clarin.eu/fcs/diagnostic/4diagnostic, i.e. add the appropriate XML fragment to the <sru:diagnostics> element of the response. The Endpoint MAY treat this condition as fatal, i.e. simply issue the diagnostic and perform no search, or it MAY treat it a non-fatal and perform the search.

# Normative Appendix

## List of extra request parameters

The following extra request parameters are used in CLARIN-FCS. The column SRU operations lists the SRU operation, for which this extra request parameter is to be used. Clients MUST NOT use the parameter for an operation that is not listed in this column. However, if a Client sends an invalid parameter, an Endpoint SHOULD issue a fatal diagnostic "Unsupported Parameter"  
(info:srw/diagnostic/1/8) and stop processing the request. Alternatively, an Endpoint MAY silently ignore the invalid parameter.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **SRU operations** | **Allowed values** | **Description** |
| x-fcs-endpoint-description | explain | true; all other values are reserved and MUST not be used by Clients | If the parameter is given (with the value true), the Endpoint MUST include an Endpoint Description in the <sru:extraResponseData> element of the explain response. |
| x-fcs-context | searchRetrieve | A comma-separated list of persistent identifiers | The Endpoint MUST restrict the search to the resources identified by the persistent identifiers. |
| x-fcs-dataviews | searchRetrieve | A comma-separated list of Data View identifiers | The Endpoint SHOULD include the additional need-to-request type Data Views in the response. |

## List of diagnostics

Apart from the SRU diagnostics defined in [OASIS-SRU-12, Appendix C] and [LOC-DIAG], the following diagnostics are used in CLARIN-FCS. The "Details Format" column specifies what SHOULD be returned in the details field. If this column is blank, the format is "undefined" and the Endpoint MAY return whatever it feels appropriate, including nothing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier URI** | **Description** | **Details Format** | **Note** |
| http://clarin.eu/fcs/diagnostic/1 | Persistent identifier passed by the Client for restricting the search is invalid | The offending persistent identifier. | If more than one invalid persistent identifiers were submitted by the Client, the Endpoint MUST generate a separate diagnostic for each invalid persistent identifier. |
| http://clarin.eu/fcs/diagnostic/2 | Resource set too large. Query context automatically adjusted. | The persistent identifier of the resource to which the query context was adjusted. | If an Endpoint limited the query context to more than one resource, it MUST generate a separate diagnostic for each resource to which the query context was adjusted. |
| http://clarin.eu/fcs/diagnostic/3 | Resource set too large. Cannot perform Query. |  |  |
| http://clarin.eu/fcs/diagnostic/4 | Requested Data View not valid for this resource. | The Data View MIME type. | If more than one invalid Data View was requested, the Endpoint MUST generate a separate diagnostic for each invalid Data View. |

# Non-normative Appendix

The following sections are non-normative.

## Syntax variant for Handle system Persistent Identifier URIs

Persistent Identifiers from the Handle system are defined in two syntax variants: a regular URI format for the Handle protocol, i.e. with a hdl: prefix, or actionable URIs with a http://hdl.handle.net/ prefix. Generally, CLARIN software should support both syntax variants, therefore the CLARIN-FCS Interface Specification does not endorse a specific syntax variant. However, Endpoints are recommended to use the actionable syntax variant.

## Referring to an Endpoint from a CMDI record

Centers are encouraged to provide links to their CLARIN-FCS Endpoints in the metadata records for their resources. Other services, like the VLO, can use this information for automatically configuring an Aggregator for searching resources at the Endpoint. To refer to an Endpoint, a <cmdi:ResourceProxy> element with child-element <cmdi:ResourceType> set to the value SearchService and a @mimetype attribute with a value of application/sru+xml need to be added to the CMDI record. The content of the <cmdi:ResourceRef> element must contain a URI that points to the Endpoint web service.

<cmdi:CMD xmlns:cmdi="http://www.clarin.eu/cmd/" CMDVersion="1.1">

<cmdi:Header>

<!-- ... -->

<cmdi:MdSelfLink>http://hdl.handle.net/4711/0815</cmdi:MdSelfLink>

<!-- ... -->

</cmdi:Header>

<cmdi:Resources>

<cmdi:ResourceProxyList>

<!-- ... -->

<cmdi:ResourceProxy id="r4711">

<cmdi:ResourceType mimetype="application/sru+xml">↩  
 SearchService</cmdi:ResourceType>

<cmdi:ResourceRef>  
 http://repos.example.org/fcs-endpoint</cmdi:ResourceRef>

</cmdi:ResourceProxy>

<!-- ... -->

</cmdi:ResourceProxyList>

</cmdi:Resources>

<!-- ... -->

</cmdi:CMD>

## Endpoint custom extensions

The CLARIN-FCS protocol specification allows Endpoints to add custom data to their responses, e.g. to provide hints to an (XSLT/XQuery) application that works directly on CLARIN-FCS. It could use the custom data to generate back and forward links for a GUI to navigate in a result set.

The following example illustrates how extensions can be embedded into the Result Format:

<fcs:Resource xmlns:fcs=http://clarin.eu/fcs/resource  
 pid="http://hdl.handle.net/4711/0815">

<fcs:DataView type="application/x-clarin-fcs-hits+xml">

<hits:Result xmlns:hits="http://clarin.eu/fcs/dataview/hits">

The quick brown <hits:Hit>fox</hits:Hit> jumps over the lazy↩  
 <hits:Hit>dog</hits:Hit>.

</hits:Result>

</fcs:DataView>

<!--

NOTE: this is purely fictional and only serves to demonstrate how

to add custom extensions to the result representation

within CLARIN-FCS.

-->

<!--

Example 1: a hypothetical Endpoint extension for navigation in a result

set: it basically provides a set of hrefs, that a GUI can convert into

navigation buttions.

-->

<nav:navigation xmlns:nav="http://repos.example.org/navigation">

<nav:curr href="http://repos.example.org/resultset/4711/4611" />

<nav:prev href="http://repos.example.org/resultset/4711/4610" />

<nav:next href="http://repos.example.org/resultset/4711/4612" />

</nav:navigation>

<!--

Example 2: a hypothetical Endpoint extension for directly referencing  
 parent resources: it basically provides a link to the parent resource,  
 that can be exploited by a GUI (e.g. build on XSLT/XQuery).

-->

<parent:Parent xmlns:parent="http://repos.example.org/parent"

ref="http://repos.example.org/path/to/parent/1235.cmdi" />

</fcs:Resource>

## Endpoint highlight hints for repositories

An Aggregator can use the @ref attributes of the <fcs:Resource>, <fcs:ResourceFragment> or <fcs:DataView> elements to provide a link for the user to directly jump to the resource at a Repository. To support hit highlighting, an Endpoint can augment the URI in the @ref attribute with query parameters to implement hit highlighting in the Repository.

In the following example, the URI http://repos.example.org/resource.cgi/<pid> is a CGI script that displays a given resource at the Repository in HTML format and uses the highlight query parameter to add highlights to the resource. Of course, it's up to the Endpoint and the Repository, if and how they implement such a feature.

<fcs:Resource xmlns:fcs=http://clarin.eu/fcs/resource  
 pid="http://hdl.handle.net/4711/0815">

<fcs:DataView type="application/x-clarin-fcs-hits+xml"  
 ref="http://repos.example.org/resource.cgi/4711/0815?highlight=fox">

<hits:Result xmlns:hits="http://clarin.eu/fcs/dataview/hits">

The quick brown <hits:Hit>fox</hits:Hit> jumps over the lazy dog.

</hits:Result>

</fcs:DataView>

</fcs:Resource>

1. <http://centres.clarin.eu/> [↑](#footnote-ref-1)