A brief summary of the Z39.50 architecture and functionality (Z39.50 services).

# 3. Information Retrieval Service

The Information Retrieval service definition describes an activity between two applications: an initiating application, the client, and a responding application, the server. The server is associated with one or more databases.

Communication between the client and server is carried out by the Z39.50 protocol. The specification is logically divided into procedures pertaining to the client and procedures pertaining to the server.

## 3.1 Model and Characteristics of the Information Retrieval Service

Communication between client and server is via a Z39.50-Association (Z-association). A Z-association is explicitly established by the client and may be explicitly terminated by either client or server, or implicitly terminated by loss of connection.

There may be multiple consecutive Z-associations for a connection. There may be multiple consecutive as well as concurrent operations (see 3.1.2) within a Z-association.

The roles of client and server may not be reversed within a Z-association. A Z-association cannot be restarted, thus once a Z-association is terminated no status information is retained, except information that is explicitly saved.

The service definition describes services and operations; models for these are described in 3.1.1 and 3.1.2. Services are grouped by facilities; the Z39.50 facilities and services are defined in 3.2.

### 3.1.1 Z39.50 Services

Z39.50 services are carried out by the exchange of messages between the client and server. A message is a request or a response. Services are defined to be confirmed, non-confirmed, or conditionally-confirmed. A confirmed service is defined in terms of a request (from the client or server) followed by a response (from the peer). For example, Search is a confirmed service, initiated by the client; the Search service is defined in terms of a Search request from the client followed by a Search response from the server. Access-control is an example of a confirmed service initiated by the server.

A non-confirmed service is defined in terms of a request from the client or server, with no corresponding response. For example, TriggerResourceControl is a non-confirmed service initiated by the client; Segment is a non-confirmed service initiated by the server.

A conditionally-confirmed service is a service that may be invoked as either a confirmed or non-confirmed service. It is defined in terms of a request (from the client or server) followed possibly by a response (from the peer). For example, Resource-control is a conditionally-confirmed service, initiated by the server.

### 3.1.2 Z39.50 Operations

This standard describes nine operation types: Init, Search, Present, Delete, Scan, Sort, Resource-report, Extended-services, and Duplicate Detection.

A request from the client of a particular operation type initiates an operation of that type (for example a Search request initiates a Search operation) which is terminated by the respective response from the server. Only the client may initiate an operation, and not all client requests do so (see 3.4).

A request that initiates an operation is called an initiating request and a response that ends an operation is called a terminating response. From the client perspective, an operation begins when it issues the initiating request, and ends when it receives the terminating response. From the server perspective, the operation begins when it receives the initiating request and ends when it sends the terminating response. An operation consists of the initiating request and the terminating response, along with any intervening related messages (see 3.4).

### 3.1.3 Model of a Database

The objective of this standard is to facilitate interconnection of clients and servers for applications where clients search and retrieve information from server databases. The ways in which databases are implemented differ considerably; different systems have different styles for describing the storage of data and the means by which it can be accessed. A common, abstract model is therefore used in describing databases, to which an individual system can map its implementation. This enables different systems to communicate in standard and mutually understandable terms, for the purpose of searching and retrieving information from a database. The search and retrieval models are described in 3.1.4 and 3.1.5.

The term database, as used in this standard, refers to a collection of records. Each record is a collection of related information. The term database record refers to a local data structure representing the information in a particular record. Associated with a database are one or more sets of access points that can be specified in a search for database records (see 3.1.4), and one or more sets of elements that may be retrieved from a database record (see 3.1.5). An access point is a unique or non-unique key that can be specified (either alone or in combination with other access points) in a search for records. An access point may or may not correspond to one or more elements (defined by an abstract syntax), For example, the (abstract) access point “title” might be used to search a particular database and might correspond to the Main Title data element for that database. The same access point might be used to search a different database, and for that database it might correspond to the Series Title data element.

### 3.1.4 Searching a Database

A query is applied to a database, specifying values to be matched against the access points of the database. The subset of records formed by applying a query is called the result set (see 3.1.6). A result set may itself be referenced in a subsequent query and manipulated to form a new result set.

A search request specifies one or more databases and includes a query. The type-1 query defined in this standard (see 3.7) consists of either a single access point clause, or several access point clauses linked by logical operators. For example:

In the database named "science fiction" find all records for which 'title' contains "galaxy" AND 'author' contains "adams". ( " 'title' contains "galaxy" " is an access point clause, as is " 'author' contains "adams" ". "AND" is a logical operator.)

Each access point clause consists of a search term and attributes. The attributes qualify the term; usually, one of the attributes corresponds to a normalized access point, against which the term (as qualified by the other attributes) is matched. Each attribute is a pair representing an attribute type and a value of that type (for example, type might be "usage" and value "author"; or type might be "truncation" and value "left").

Each attribute is qualified by an attribute set id, which identifies the attribute set to which the attribute belongs. An attribute set specifies a set of attribute types, and for each, a list of attribute values.

### 3.1.5 Retrieving Records from a Database

Following the processing of a search, the result set is available at the server, for reference by the client, for subsequent searches or retrieval requests. When requesting the retrieval of a record from a result set, the client may supply a database schema identifier, element specification, and record syntax identifier.

For the purpose of retrieving records from a result set, associated with each database are one or more schemas. A schema represents a common understanding shared by the client and server of the information contained in the records of the database, to allow the subsequent selection of portions of that information via an element specification.

A schema defines an abstract record structure which, when applied to a database record results in an abstract database record, which is an abstract representation of the information in the record. An element specification applied to an abstract database record result in another instance of the abstract database record (the latter may be a null transformation). The element specification selects elements from the abstract database record, and may also specify variant forms for those elements.

The server applies a record syntax to an abstract database record, resulting in an exportable structure referred to as a retrieval record.

### 3.1.6 Model of a Result Set

Logically, a result set is an ordered list of items, each of which is a pointer to a database record; it is used as a selection mechanism for the transfer of database records identified by a query. A result set itself is considered to be a purely local data structure and is not transferred (that is, records are transferred, but not the local pointers to the records).

In general, it is assumed that query processing does not necessarily require physical access to records; a result set is thus assumed to be the identification of (e.g., pointers to) records, as opposed to the actual set of records, selected by a query.

It is important to distinguish the physical implementation from the abstract model. How a server chooses to implement result sets is an implementation matter; a result set may be a copy of the database records, a table of pointers, or there may not even be a physical result set (the server might execute the query every time the result set is referenced, package up and send the requested records, and otherwise immediately discard the results; however, the result set model does require that a result continue to refer to the same records in the same order). But from the client point of view (and the abstract model) the result set is a set of items, or vectors, where each includes a database name and a pointer to a record within the database.

#### 3.1.6.1 Concurrency and Update Considerations

It is not assumed that the database records are locked. Methods of concurrency control, which would prevent modification or deletion of result set records, are not addressed by this standard.

The server could potentially modify a record pointed to by a result set. For example suppose the client requests record 1, and then subsequently requests record 1 again; the second time, the record may have changed. There is no direct mechanism provided by Z39.50 to prevent this (though Extended Services might be used to lock a record). When a client does modify a record referenced by a result set, then when the client subsequently requests the record, the server might refuse to supply it and instead supply a surrogate diagnostic, but there is no such requirement.

A successful search (see 3.2.2.1.10) results in the creation of a result set. However, a server may unilaterally delete a result set at any time for no specified reason; the next time that the client attempts to refer to  that result set (for example in a Present request) the server might send a diagnostic to the effect "result set does not exist". (The server may instead send a diagnostic to the effect that "the result set no longer exists because it was unilaterally deleted", however, the server is not obliged to do so and may simply take the position that the result set never existed.) Therefore, although in the abstract a successful search results in the creation of a result set, as a practical matter, if a server does not actually create a result set, it is not violating the Z39.50 protocol.

#### 3.1.6.2 Order of Result Set Records

The records in a result set are not necessarily ordered according to any specific or predictable scheme (although, whatever the order, it is assumed static). Thus assume for example there is a result set of 100 records, and the client wants the three most recent (i.e. based for example on 'date of publication'). There is no simple way to find those records, short of retrieving all the records in the result set. The Z39.50 Sort service however allows the client to request that a particular result set be sorted based on a specific key (e.g. ‘date of publication’, descending). If the server supports the sort service (and also supports sorting on the requested key, in this example, ‘date of publication’) then following the Sort, the client may subsequently retrieve the first three records, and they will be the three most recent.

#### 3.1.6.2 Logical Structure of a Result Set

For the purpose of retrieving records, the logical structure of a result set is that of a named, ordered list of items. Each item is a triple consisting of:

(a) An ordinal number corresponding to the position of the triple in the list

(b) A database name

(c) A unique identifier (of local significance only) of a record within the database named in (b)

A result set item is referenced by its position within the result set, that is, by (a).

For the purpose of searching, when a result set is used as an operand in a query, the logical structure is one of the following:

| | |
|---|---|
| Basic model | A set of pairs, each consisting of (b) and (c) of the above model for retrieval. |
| Extended model | A set of triples, each consisting of (b) and (c) of the retrieval model; and including unspecified information associated with each record, which may be used as a surrogate for the search that created the result set. |

**Note:** Query specifications may indicate that the basic model applies, or under what condition the extended model applies and the nature of the unspecified information. For the type-1 query, when version 2 is in effect, the basic model applies.

### 3.1.7 Model of Extended Services

The family of Z39.50 services includes the Extended Services (ES) service. "Extended services" refers to a class of services recognized by this standard, but which are not Z39.50 services (as described in 3.1.1). The ES service *is* a Z39.50 service, and an ES operation results in the initiation of an extended services task. The *task* is *not* considered part of the Z39.50 ES operation.

An ES operation is initiated by the client, via an ES request. The ES response, which completes the operation, does not (necessarily) signal completion of the task; it may indicate for example that the task has started or is queued (or it might indicate that the task has been completed; in fact the ES request may specify that the task should be completed prior to the ES response). An ES task may have a lifetime beyond the Z-association.

Examples of extended services are: saving a result set or query, and exporting or ordering a document.

Each ES task is represented by a database record, called a task package, maintained by the server in a special database, the "extended services database". The client uses the ES request to cause creation of a task package on the ES database. The database may be searched, and records retrieved, by the Z39.50 Search and Retrieval facilities. The client may search for packages of a particular type, or created by a particular user, or of a particular status (i.e. pending, active, or complete), or according to various other criteria. In particular, the client may search the database after submitting an ES request (during the same or a subsequent Z-association), for a resulting task package, to determine status information pertaining to the task, for example, to determine whether the task has started.

### 3.1.8 Explain

The client may obtain details of the server implementation, including databases, attribute sets, diagnostic sets, record syntaxes, and element specifications supported. The client obtains these details through the Z39.50 Explain facility. The server maintains this information in a database that the client may access via the Z39.50 Search and Present facilities.

This "explain" database appears to the client as any other database supported by the server, but it has a well-known name and a pre-defined record syntax. Also, certain search terms, corresponding to information categories, are predefined in order to allow a semantic level of interoperability. Each information category has its own record layout, and all are included in the Explain syntax.

## 3.2 Facilities of the Information Retrieval Service

Sections 3.2.1 through 3.2.11 describe the eleven facilities of the Information Retrieval service. Most consist of logical groups of services; in several cases, a facility consists of a single service. Additional services may be added to any facility in future versions of this standard. Following is a summary description of the eleven facilities.

| | |
|---|---|
| **Initialization Facility** | Init Service: A confirmed service invoked by the client to initiate an Init operation. |
| **Search Facility** | Search Service: A confirmed service invoked by the client to initiate a Search operation. |
| **Retrieval Facility** | The Retrieval facility consists of two services:<br>– Present Service: A confirmed service invoked by the client to initiate a Present operation.<br>– Segment Service: A non-confirmed service initiated by the server, during a Present operation.<br>**Note:** a Present operation thus consists of a Present request followed by zero or more Segment requests followed by a Present response. |
| **Result-set-delete Facility** | Delete Service: A confirmed service invoked by the client to initiate a Delete operation. |
| **Browse Facility** | Scan Service: A confirmed service invoked by the client to initiate a Scan operation. |
| **Sort Facility** | The Sort facility consists of two services:<br>– Sort Service: A confirmed service invoked by the client to initiate a Sort operation.<br>– Duplicate Detection Service: A confirmed service invoked by the client to initiate a Duplicate Detection operation. |
| **Access Control Facility** | Access-control service: A confirmed service initiated by the server. It does not initiate an operation, and it might or might not be part of an active operation. |
| **Accounting/Resource Control Facility** | The Accounting/ Resource Control facility consists of three services:<br>– Resource-control Service: A conditionally-confirmed service initiated by the server. It does not initiate an operation, and it might or might not be part of an |

| | |
|---|---|
| | active operation.<br>– Trigger-resource-control Service: A non-confirmed service initiated by the client during an operation.<br>– Resource-report Service: A confirmed service invoked by the client to initiate a Resource report operation. |
| **Explain Facility** | The Explain facility does not include any services, but uses the services of the Search and Retrieval facilities. |
| **Extended Services Facility** | Extended-services Service: A confirmed service invoked by the client to initiate an Extended-services operation. |
| **Termination Facility** | Close Service: A confirmed service initiated by the client or server. It does not initiate nor is it part of any operation. It allows a client or server to abruptly terminate all active operations and to initiate termination of the Z-Association. (Following termination of the Z-Association the client may subsequently attempt to initialize another Z-Association using the Init service.) |