ENABLING CSCW SYSTEMS TO AUTOMATICALLY BIND EXTERNAL KNOWLEDGE BASES

Thomas Bopp and Jonas Schulte
Heinz Nixdorf Institute, University of Paderborn, Fürstenallee 11, 31102 Paderborn, Germany

Thorsten Hampel
Department of Knowledge and Business Engineering, University of Vienna, Rathausstr. 19/9, A-1010 Vienna, Austria

Keywords: Digital Library, Integrated Search, Distributed Knowledge Spaces, CSCW, Automatic Binding.

Abstract: The usage of CSCW systems for teaching, training, and research collaboration increases constantly, as it offers a time- and place-independent, as well as a cost-effective platform. The user's search should not be restricted to local material; in fact, users benefit from different search environments, as, for example digital libraries to find appropriate working material. Searching and further processing of documents imply a media breach since the search cannot be invoked in current CSCW systems directly. This paper presents the first prototype of a CSCW system which enables users to search in external sources without media breach. To provide arbitrary search environments no restrictions to data formats or search functionalities are allowed. Hence we have enhanced search environments with self description capabilities in order to realize an automatic binding of search environments in CSCW systems. By search environments we address any system offering searchable knowledge bases, such as digital libraries or the CSCW system itself. Furthermore our concept supports local search and searching in different external sources at the same time.

1 INTRODUCTION

CSCW (Computer Supported Cooperative Work) systems are an efficient solution to establish cooperative work independent from the local places of participants. Their fields of application are not limited to universities, as it is also beneficial to the industry in regard to employee training. In many applications, as for example structuring cooperative work processes or field of employee trainings, it is essential to have a reliable database to work with. Usually, not all data needed for arbitrary trainings and research/teaching activities is already contained within the CSCW system. Accordingly, users search for information in the internet as well as in digital libraries. The credibility of external information is of crucial importance, which is often not given by freely accessible data from the web.

In contrast the credibility of information available from digital libraries is generally undoubted and therefore digital libraries are an excellent data source for reliable documents, which furthermore can be accessed independently from time and place. Enabling publication of and access to information in a digital format has already been realized by digital libraries such as DSpace(Smith, 2002), Fedora (Payette and Lagoze, 1998), and DuEPublico (Gollan et al., 1999). There is a significant trend of providing access to information in a digital way which enables an easy further processing as well as time and place independent utilization.

The goal of our approach is to develop mechanisms which allow for a flexible integration of different knowledge sources into any collaborative systems. In this context, the new generation of web services are of central importance in regard to our mistel-approach. In contemporary digital libraries the search is often invoked by users via a web-interface. To utilize material from digital libraries in CSCW systems, there are currently two steps to be effected: First of all, the user has to search via the web-interface of the digital library. Afterwards, the appropriate results are added into the content of the CSCW system. In most cases, this requires to buffer the files from the result page of the digital library and then to upload them to the CSCW system. It is obvious that this

Bopp T., Schulte J. and Hampel T. (2007).
ENABLING CSCW SYSTEMS TO AUTOMATICALLY BIND EXTERNAL KNOWLEDGE BASES.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - DISI*, pages 323-329
DOI: 10.5220/0002382603230329

process is not only inefficient and not user-friendly, but also slows down the establishment of CSCW systems and digital libraries. However, performing these steps is the only possible way at the moment to use data from digital libraries in CSCW systems. Our objective is to avoid this media breach between searching in external sources and using founded material in CSCW systems in order to boost the establishment of both platforms as well as their utilization. A solution without media breach would require the user being able to invoke search functionalities of digital libraries *within* the CSCW system. To enable other systems to use their search functionalities, some digital libraries already provide appropriate interfaces by deploying web services (Petinot et al., 2004; Gollan et al., 1999). However, no contemporary CSCW system benefits from this possibility.

In this paper we address the usage and search of external information in CSCW systems without media breach. We not only describe the integration of external search functionalities in a CSCW system, in fact it is obvious that an automatic binding of digital libraries/repositories in CSCW systems is definitely more beneficial than a manual integration. To enable a complete automatic binding process, several requirements for the search environments have been identified. As an example implementation we have integrated the DuEPublico binding in the CSCW system open sTeam (Bopp et al., 2006b). Here, a detailed view of the realized concept is presented in order to point out the advantages of an integrated search functionality. Our approach is not limited to the mentioned demonstrational systems. Instead, it is a open web service implementation which can be added easily to any collaborative system and digital library.

Our architecture combines local and external search resources offering extended search capabilities to the user. In this regard, we have identified a number of scenarios (see section 1.1): A user might search for a search term in all possible resources or explicitly choose an external resource. Regarding the concrete example of the object oriented open sTeam CSCW system, external and local search results are usually imported into knowledge spaces and can be arranged using the room and spatial metaphors in order to make appropriate search results available for other CSCW users.

This paper is organized as follows: related work is presented in section 2, while the description of our developments is divided according to requirements in section 3. Next, a detailed view of the system interaction is given in section 4. Before concluding our results the adjusted concept for searching in parallel in different search environments is explained in section 5.

1.1 Motivating Scenarios

In this section three different scenarios are presented which point out the various search requirements for a user-friendly CSCW system. Flexible search functionalities as the ones described in this paper, are quite new to common CSCW systems. Our analysis presents a concept to realize these three scenarios in CSCW systems mentioned below. A basic framework and first prototypes developed on its basis will be introduced, which support searching according to the different scenarios and offer to invoke external search functionality of one digital library.

Scenario 1: Local Search

A user searches for local objects, as, for example a specific document, group, user, or knowledge space. If the user searches for a certain group to request membership, he already knows the local availability of the group and searching in local resources is sufficient.

Scenario 2: Literature Research

This scenario describes the research for literature in digital libraries. A user searches for general keywords or more specific meta data such as authors, conferences, or book titles. Digital libraries/repositories offer a large number of articles and provide rich meta data for the entries. Usually, not only a single repository is searched by the user, but also literature in the established digital libraries of ACM, IEEE, and CiteSeer are scanned for results. Therefore, within a CSCW system the user would in this case choose several external resources to which the search request should be sent.

Scenario 3: General Searching

A user might have no precise idea of where to find something. Therefore, he searches without further specifications all available resources, which possibly might lead to a large number of search results.

2 RELATED WORK

Digital libraries such as DSpace (Smith, 2002), Fedora (Payette and Lagoze, 1998), and DuEPublico (Gollan et al., 1999) provide digital access to their inventories, as well as web services to invoke their search functionalities remotely. To enable searching, it is necessary to identify and collect metadata from documents.

The Open Archives Initiative provides the OAI-PMH (Lagoze and de Sompel, 2001) standard for

harvesting metadata of documents. This approach is meant to be a low-barrier solution with roots in enhancing access to e-print archives. The need for a common metadata format has been identified before and resulted in the development of the Dublin Core Metadata Element Set (Weibel, 1998). This set became an ISO standard and most digital libraries like DSpace, Fedora, and DuEPublico, provide the data using Dublin Core (DC) elements. The prototype presented in this paper bases on the awareness that arbitrary search environments do not provide an uniform metadata format. Accordingly, an automatic binding requires a search environment's ability to describe their functionalities and formats on their own.

Since the DC elements are designated to be used for metadata description, it lacks description terms for service functionalities and for additional information required to perform searches. Therefore, the ZeeRex format (ZeeRex, 2004) can be used to describe service functionalities. In the process of an automatic binding the self description method of a search environment returns a ZeeRex document, which is evaluated from the CSCW system in order to enable the user to invoke the search functionalities without media breach. There is no contemporary CSCW system which offers the search in external sources without media breach. The developed prototype extends the open sTeam (Bopp et al., 2006b) CSCW system.

Another approach to standardize the access of search functionalities is the standard search protocol Search/Retrieve Web Service (SRW, 2005), which includes the search, scan, and explain operations. However, this standard is not (fully) supported by arbitrary search environments and rather does not define the exchange formats of the metadata. That is why even if SRW is supported, an automatic binding requires the search environment's capability of self description.

3 REQUIREMENTS FOR AUTOMATIC BINDING

In this section the requirement analysis for automatically integrating search functionalities of external search environments into a CSCW system is presented. Since the requirements result from the process of using external search functionalities, section 3.1 explains the workflow of the search invoking. The resulting requirements for the search web service are described in section 3.2.

Our recent conceptual approach in the field of Computer Supported Cooperative Work and Learning (CSCW/L) has been to build cooperative knowledge spaces. One resulting implementation from this approach is the client/server system called open sTeam (Bopp et al., 2006b). The idea of this system is to combine document management facilities with a room metaphor in collaborative knowledge spaces. This concept is especially advantageous since one user can perform a search, set results in relation with already existing documents, and make them available for other users in the distributed knowledge space.

Our development in the CSCW system open sTeam provides access to search functionalities of arbitrary digital libraries and additionally, of any search environment which deploys adequate web services. The identified requirements are independent from the CSCW system, which should interact with the search environment. Hence, our results can be transferred to extend other CSCW systems in a similar manner.

3.1 General Search Process

Distributed knowledge spaces are containers for information from various external data sources. Accordingly, the content stored in the CSCW system bases on external information and the search functionalities usable in open sTeam should not be limited to one (external) search environment. In our solution the user can select several search environments the web services of which are invoked, which conforms to the principle of meta search engines. Hence, the first step for the user is to select the search environments to which his or her search request is sent. As a second step, the user specifies his or her search request in an adequate search dialog in the web-interface of open sTeam. Please note that the presented search dialog depends on the selected search environments. Search dialogs for several search environments can differ since they may support different indexed search fields like keywords, author, title, etc. After the user has submitted the search dialog, open sTeam generates search requests conform with the selected environments. The web services are invoked and the results are presented in the web-interface of open sTeam. Since the results may be retrieved from different environments, an aggregation of the result hits is required (Bopp et al., 2006a). A result page provides the opportunity to add the found documents directly to a distributed knowledge space for sharing them with other participants. The system therefore offers not only a standardized frontend to different knowledge sources, but allows the flexible integration of new knowledge source in various ways.

3.2 Functionalities of Search Web Services

To provide highest flexibility, it is essential to couple new search environments with the CSCW system without additional effort. However, each search environment can arbitrarily define the supported indexed search fields and configuration options. Hence, it is necessary that web services from a search environment (web service provider) are able to explain themselves. Only if explanation methods are supported, CSCW systems as web service consumers can configure themselves automatically, build dynamically search dialogs, and automatically generate adequate search requests. The new approach of our development is to divide the usage of a search web service into the following three steps: First of all a connection with an optional authentication process has to be established. Afterwards the CSCW system and the search environment negotiated the interaction process, which requires the search web service to explain its functionalities. Finally, the CSCW system is able to use the underlying web service. Figure 1 illustrates this proc

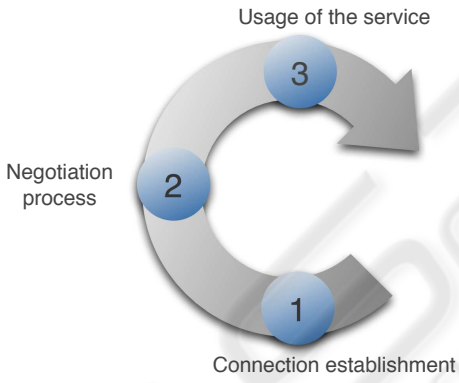

Figure 1: Usage of Search Web Services.

In this section we focus on step 2 - the negotiation process between the CSCW system and the search environment. To provide highest usability of the description of the search web service's functionalities a simple and customizable exchange format is required. Therefore we use the ZeeRex format (ZeeRex, 2004) which extends the Z39.50 standard maintained by the Library of Congress. ZeeRex is an attempt to define an "Explain"-like mechanism for search environments. If a digital library supports this explain mechanism, a CSCW system can invoke the explain method to retrieve an XML document that describes in details the search environment which is required for an automatic binding. According to the ZeeRex DTD the XML document for explaining a search environment has the following structure:

```
<explain>
   <serverInfo> ... </serverInfo>
   <databaseInfo> ... </databaseInfo>
   <metaInfo> ... </metaInfo>
   <indexInfo> ... </indexInfo>
   <schemaInfo> ... </schemaInfo>
</explain>
```

As described above, a ZeeRex document contains, at most five sections surrounded by an `explain` tag. The `serverInfo` section contains basic information required for connecting to the database such as protocol, host, port, and database name. We focus on the `indexInfo` and the `schemaInfo` sections. The `schemaInfo` section refers to available XML schemata useable for returning results. Knowing the format of the results is necessary to transform them for the final presentation in open sTeam. In the `indexInfo` section(s) all usable ways in which the server may be interrogated are recorded. These are described by using `<index>` elements. The `<index>` element is a representation of a single type of search. For example, a title search is one index and an author search is another index. Each phrase of a search request belongs to exactly one search index. Furthermore, the search request may only contain at most one phrase for one search index.

We have developed an example ZeeRex document explaining the DuEPublico search environment which is presented in section 3.2.1. In section 4 we describe how to evaluate ZeeRex documents and particularly `<index>` elements to automatically build search dialogs in CSCW systems. Please note that the explain functionality is not yet supported by DuEPublico. In our prototype the information has to be integrated manually. To enable CSCW systems to bind search environments completely automatic, it is mandatory to support the explain mechanism.

3.2.1 Zeerex Description of Duepublico

As described previously, the focus of this section is on interpreting the `<indexInfo>` parts of the ZeeRex description, since they present the possible search types. DuEPublico search environments offer a plurality of search indexes and selection fields, which can be used in a search request. Furthermore, configuration fields enable the customization of the result presentation. The search web service supports the same functionalities as searching via the DuEPublico web-interface.

For describing the indexed search fields "title" and "author" the explain document contains the following `<index>` element.

```
<indexInfo>
  <index id="title" sort="true" search="true">
    <title>Document Title</title>
```

```
    <map>
      <name set="dc">title</name>
    </map>
  <index>
  <index id="name" sort="true" search="true">
    <title>Author/Creator</title>
    <map primary="true">
      <name set="dc">author</name>
    </map>
    <map>
      <name set="dc">creator</name>
    </map>
  <index>
</indexInfo>
```

The indexed search field title can be used in search requests and is one of the possible sort criterions. Because of these supported functionalities for the search field, the `<index>` element has the attributes `search` and `sort`. The `<title>` element describes the search index, which is here the document title. Search requests sent to the web service have to be conform with one of the supported format of the search environment. Each search index is mapped to a specific term in this format. By enabling search environments to support various request formats, more than one `<map>` element can be defined. The relation of the `<map>` element to one format is given by its attribute `set`. In the example mentioned above DuEPublico accepts only search requests in the Dublin Core (DC) format (Weibel, 1998). Please note, as the example show it for the author, different Dublin Core elements can be mapped to the same search index.

The `<index>` elements for selection fields and the configuration fields should be described similar to the search fields. The configuration fields determine constraints for the reply of the search web service, as for example, the maximum number of results. It is common practice to characterize materials in a library's inventory by classifications. A media type of the material, for example, is determined like animation, audio file, figure, text, etc. The ZeeRex DTD lacks describing selection and configuration fields. That is why we extend the ZeeRex document as follows:

```
<explain xmlns:sEnv=
"http://systemkonvergenz.de/ZeeRexExtension">
  ...
  <indexInfo>
    <index id="format" search="true">
      <title>Media Type</title>
      <map>
        <name set="dc">format</name>
        <sEnv:method>
          <sEnv:name>
            doListClassification
          </sEnv:name>
          <sEnv:params>
            format
          </sEnv:params>
        </sEnv:method>
      </map>
    </index>
    <index id="maxHits" search="false">
      <title>Maximum Hits</title>
      <map>
        <name set="conf">maxResults</name>
        <sEnv:selection>
          <sEnv:item>50</sEnv:item>
          <sEnv:item>200</sEnv:item>
          <sEnv:item>1000</sEnv:item>
        </sEnv:selection>
      </map>
    </index>
    ...
  </indexInfo>
  ...
</explain>
```

The most important aspect to use selection fields is the need to read out the selectable data dynamically. To support automatic binding the method responsible for retrieving the values is defined by the usable parameters in the ZeeRex document.

4 SYSTEM INTERACTION

The first step to an integrated search functionality is the generation of an adequate search dialog. To do this the explain method has to be called by the CSCW system, which retrieves a ZeeRex document describing the supported search fields, selection fields, and configuration fields of the search environment. The following section 4.1 explains the evaluation of the retrieved ZeeRex document by the CSCW system for building the search dialog. Afterwards section 4.2 presents the process of transforming the user's inputs into a search request that can be interpreted by the addressed search environment. At last, section 4.3 provides the description of how to prepare the retrieved search results for the usage in the CSCW system.

4.1 Generating the Search Dialog

As shown in section 3.2.1, `<indexInfo>` elements contain information how to specify correct search requests. The CSCW system has to evaluate these elements in order to automatically generate a suitable search dialog. This is done by parsing the received ZeeRex document after having invoked the explain method of the search environment. It is important, not to transform every `<index>` element into a search field, since `<index>` elements are also used to describe selection and configuration fields.

The type of an `<index>` element can be identified referring to the children of the `<map>` element. According to the possible differentiation between selection, configuration, and search fields, the CSCW system can generate the search dialog as follows:

For each search field a corresponding input field will be assimilated into the search dialog. After-

wards the configuration fields will be included into the search dialog as choice fields. For every `<item>` element an adequate choice option for the generated choice field is inserted. Selection fields present dynamic data for which it is not reasonable to store it in the manually configured ZeeRex document. Accordingly, a digital library should provide a method to query entries for a dynamic information. DuEPublico, for example, provides such a service for receiving classifications of the available materials (see section 3.2.1) which is called by the CSCW system. The ZeeRex document contains all information that is necessary to query the classification entries by defining the name and the parameters of the responsible method. With the newly gained information about supported classifications the CSCW system can adds choice fields with appropriate choice options into the search dialog.

4.2 Building the Search Request

The ZeeRex document is used to generate an adequate search dialog for the particular search environment described in section 4.1. It also contains information to build search requests which can be interpreted by the search environment. Since arbitrary search environments do not support the same query language, the CSCW system has to create different queries for different search environments.

The ZeeRex document defines for each field the variable to which the input has to be mapped during the search request. As described in section 3.2.1, a search term may be addressed by different terms of distinguished or equal request formats. DuEPublico only supports requests in the DC-format, which was extended by the `conf`-set for configuration options. The CSCW system has to extract the non-empty fields of the user's submitted search dialog and integrate adequate search phrases into the search request. These phrases are of the form of `fieldNames Operator userInput`. The useable field names are listed in the ZeeRex document in the `<name>`-element.

After the search request is built, the CSCW system invokes the web service to execute the search. In addition to the search request some search environments require the transmission of authentication information for the usage of the search functionality, as well as for access to the inventory.

4.3 Handling the Search Results

The reply of the search web service is an XML document which contains the results in a specific format. DuEPublico, for example, sends the results in the DC-format. If a search environment returns the results in a different format as the one used by the CSCW system in the search request, it is crucial that the CSCW system can realize the mapping of the received terms to the requested phrases with the help of the ZeeRex document (adequate `name`-elements have to exist).

The CSCW system transforms the received results with an XSLT stylesheet into a suitable presentation for the user in the CSCW system. In the transformation process additional functionalities of the CSCW system can be integrated. In the open sTeam system the backup-functionality was integrated into the result presentation in order to enable users to store found material directly in knowledge spaces.

5 PARALLEL SEARCH IN DIFFERENT ENVIRONMENTS

The scenarios presented in section 1.1 show the users' demand for performing specific searches in different search environments. Often, sending the same search request to different search environments at the same time, (including or excluding the local CSCW search) will simplify searching, making it more efficient, since search results from different search environments are available at the same time and thus, can be compared. The user-friendliest solution is to present all results on a single page, which also enables the CSCW system to sort the results according to their relevance and independent from their source.

In section 4, the handling of one search environment was described, the concept of which has to be adjusted to support parallel searches in different environments. Then the first step for the CSCW system is to present a selection of useable search environments in which the user can select the ones to which his search request should be sent. After the user has selected the search environments, an adequate search dialog is presented. This search dialog may only contain search fields, selection fields, and configuration fields which are supported by *all* selected search environments. It is crucial to present the intersection of the sets of supported fields in order to return only correct results. This will be explained in the following:

A search request should be sent, for example, to DuEPublico and Google. If the search dialog presents all the input fields supported by any of the search environments, the user may add the author information. The ordinary Google search does not support author searching. In contract, if the CSCW system adds the inserted name of the author into the keyword search, Google will find also documents in which the author is mentioned but of which he is not the author. Ac-

cordingly, the Google search results are not totally correct and listing them in the same context as the DuEPublico results will not be valid.

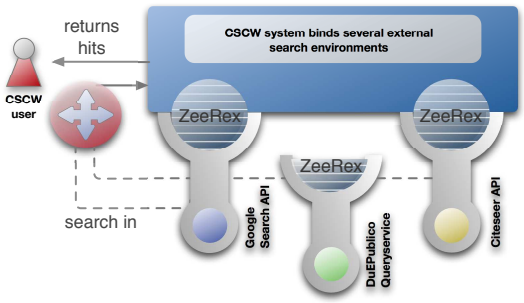

Figure 2: Binding Arbitrary Search Environments.

To generate a search dialog presenting only the intersection of the supported functionalities of several search environments, their ZeeRex documents have to be compared. Since titles and formats may differ, the intersection can only be identified if all search environments use the same `id` for equal index fields. In this case the CSCW system will be able to find equalities of search fields by comparing the `id`-attributes.

After the user has submitted the filled search dialog, the CSCW system builds adequate search request(s) for the selected environments. This step requires the generation of several search requests if the search environments support different formats. When the search results have been received by the CSCW system, adequate XSLT transformations are executed and the elements can be sorted if the required relevance information has been transmitted by the search environments or is evaluated by the CSCW system itself.

6 CONCLUSION

A user-friendly and efficient solution for searching in search environments, such as digital libraries, and for using adequate results in a CSCW system requires the avoidance of media breach. Therefore, many search environments already deploy web services for invoking their search functionalities. The presented approach developed a standardized web service to integrate different search sources through a self explanation mechanism of web services. Coupling CSCW systems with various search environments will extend the functional range of a CSCW system and support users in their individual work. As a result, an automatic binding of search environments in CSCW systems is necessary, as manual integration is too time-consuming, and therefore would limit the number of available search environments in a CSCW system will be strongly limited.

In this paper we have presented a prototype which supports automatic binding of search environments. For the automation, the search environments have to support an explain method to describe their functionalities. This method returns an individual ZeeRex document, which has been extended in this paper to satisfy the requirements for the automatic binding.

REFERENCES

Bopp, T., Hampel, T., Hinn, R., Lützenkirchen, F., Prpitsch, C., and Richter, H. (2006a). Alltagstaugliche mediennutzung erfordert systemkonvergenz in aus- und weiterbildung. In *E-Learning - alltagstaugliche Innovation?*, volume 38 of *Medien in der Wissenschaft*, pages 87–96.

Bopp, T., Hinn, R., and Hampel, T. (2006b). A service-oriented infrastructure for collaborative learning in virtual knowledge spaces. In *Education for the 21st Century: Impact of ICT and Digital Resources*, volume 210 of *International Federation for Information Processing*, pages 35–44. Springer.

Gollan, H., Luetzenkirchen, F., and Nastoll, D. (1999). Miless - a learning and teaching server for multi-media documents. In *Lecture Notes in Control and Information Sciences: Workshop on Wide Area Networks and High Performance Computing*. Springer, London.

Lagoze, C. and de Sompel, H. V. (2001). The open archives initiative: building a low-barrier interoperability framework. In *Proceedings of the 1st ACM/IEEE-CS joint Conference on Digital Libraries*, pages 54–62, New York, NY, USA. ACM Press.

Payette, S. and Lagoze, C. (1998). Flexible and extensible digital object and repository architecture (fedora). In *ECDL '98: Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 41–59, London, UK. Springer-Verlag.

Petinot, Y., Giles, C., Bhatnagar, V., Teregowda, P., and Councill, I. (2004). Citeseer-api: Towards seamless resource location and interlinking for digital libraries. In *Proceesdings of the 13th ACM Conference on Information and Knowledge Management*, pages 553–561. ACM Press.

Smith, M. (2002). Dspace: An institutional repository from the mit libraries and hewlett packard laboratories. In *ECDL '02: Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, pages 543–549, London, UK. Springer-Verlag.

SRW (2005). SRW/U - search / retrieve web services. `http://www.loc.gov/standards/sru/`.

Weibel, S. (1998). The dublin core: A simple content description format for electronic resources. *NFAIS Newsletter*, 40(7):117–119.

ZeeRex (2004). Zeerex: The explainable 'explain' service. `http://explain.z3950.org/`.