Using an application profile based service registry

Ann Apps
MIMAS, The University of
Manchester, UK
ann.apps@manchester.ac.uk

Abstract

The aim of the JISC Information Environment Service Registry (IESR) is to assist other applications, to discover and devolve materials that match their users' interests in their research, learning and teaching. This paper describes the experience of using an Application Profile throughout the application development process, from initial data design, through application implementation, to scenarios illustrating application use. Also discussed is the benefit of using an Application Profile to share both a data schema and actual data with similar initiatives.

Keywords: service registry; application profile; metadata schema.

1. Introduction

The JISC Information Environment Service Registry (IESR) (IESR, 2007a) is a middleware, shared service providing a single central registry within the JISC Information Environment (JISC, 2007). It aims to assist other applications, such as portals, virtual learning applications or research services, to discover and devolve materials that match their users' interests in their research, learning and teaching. It contains information about collections of resources that JISC (the Joint Information Systems Committee) makes available within UK Higher and Further Education, or that are useful to this community. Additionally IESR contains details of how to access technical services, both those that make the collections available, and others that play a significant role in the information environment. The early development of IESR was described in a previous Dublin Core conference paper (Apps, 2004) and details of later application developments have been published (Apps, 2006) more recently.

The design of IESR was documented using a Dublin Core Application Profile (CEN, 2003) from the outset. This paper considers the advantages of this approach and the evolution of the Application Profile (IESR, 2007b) as IESR develops into a service with identified uses.

2. The IESR Domain Model

The IESR application model comprises three types of entity: 'collection', which is an aggregation of resources; 'service', which is a system that provides one or more functions, being either an 'informational' service providing access to a collection, or a 'transactional' stand-alone service; and 'agent', which may be an owner of a collection or an administrator of a service, or both. Additionally each entity has an associated set of administrative metadata. This model is depicted in FIG. 1.

The initial design of IESR metadata was informed by the Research Support Libraries Programme (RSLP) Collection Description Schema (Powell et al., 2000), which was based on Heaney's (2000) study "An Analytical Model of Collections and their Catalogues." There are a couple of significant differences between the IESR model and the Heaney model, both made for pragmatic reasons to simplify use of the application. Within IESR the term "service", which denotes a single access point to a collection, is actually a conflation of Heaney's "location" of the collection and a "service" provided at that location. In situations where a collection is a catalogue, the catalogue and the collection it describes are conflated into a single collection description of type 'catalogue'. This means that users see, for example, a catalogue of items of type 'image',

rather than the theoretically correct but potentially confusing catalogue that describes a collection plus a second description of the collection that contains images.

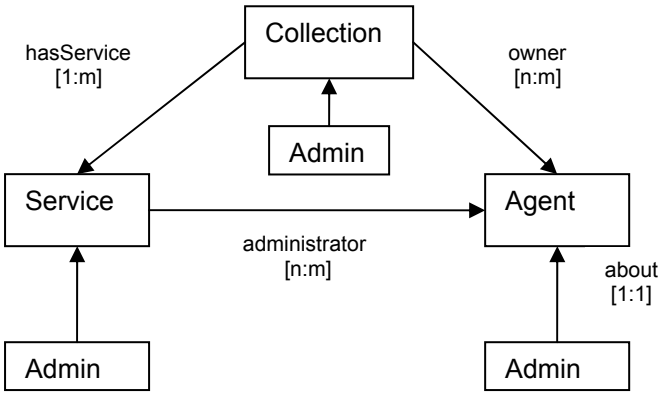

FIG. 1. The IESR domain model.

It may seem strange that a 'service registry' describes collections. And it may appear more intuitive to centre the specification around services. Some confusion arises from the various overloaded meanings of the word 'service'. Within IESR context, a service is a low level, technical access point to a data collection, or a broker service such as an OpenURL (Apps & MacIntyre, 2006) resolver (that provides access to resources appropriate for a user's context), or a component of a technical Service Oriented Architecture (Wikipedia, 2007a). In reality, within the JISC Information Environment, the function of many services is to provide access to a data collection, so some information about a collection is needed. Describing the collection as part of a service becomes complicated when there is more than one service available to access a collection, e.g. 'find' via a web search, 'find' using Z39.50 (a standard machine protocol for information retrieval) (ANSI/NISO, 1995), and an Open Archives Protocol for Metadata Harvesting (OAI-PMH) (Lagoze et al., 2004) 'harvest'. The model is much cleaner when collection-based as well as being based on the formal modelling described above. In practice the collection-based IESR model works well.

3. The IESR Application Profile

The IESR Application Profile documents the IESR metadata, which is a set of properties used to describe each entity. It defines constraints such as which properties are mandatory and which may be repeated. It is based on the Dublin Core Application Profile (DCAP) Guidelines (CEN, 2003), but with some variations. Mainly for consideration of the document length, only those application profile attributes necessary to define each property are included. The attributes 'type of term', 'refines', 'refined by' and 'similar to' are not used at all because, although they would define theoretical characteristics of the metadata properties, they are not necessary to the application. In the first version of the Application Profile the source definitions of and comments about properties were not included, again to reduce the size of the document. However these have now been included because it seems instructive to be able to view the source and IESR definitions in one place.

Various additional application profile attributes have been included, because it seems more efficient to include all documentation of the metadata properties in one place. The first extra attribute introduced was 'IESR Searchable'. This indicates which properties are searchable via the IESR application's machine interface, and documents the associated search fields corresponding to the property. Further IESR application profile attributes have since been included to provide different levels of compliance as described below in Section 5. An example of an entry in the Application Profile for a single property is shown in FIG. 2.

The DCAP Guidelines describe an application profile that captures a single entity. This corresponds to a single resource description within the Dublin Core Abstract Model (DCAM) (Powell et al., 2005), which specifies a flat set of properties for a single resource, with no provision for any composite properties according to any hierarchical model and syntax. Thus some extension of the DCAP has been required to capture the four entities of IESR. Dissemination of IESR metadata comprising several entity descriptions corresponds to a "description set" within the DCAM. In order to accommodate the several entities of IESR the DCAP is composite, split into several sections, one for each entity, preceded by a section that specifies the entities. Within the metadata itself there are properties that capture the relations between entities ("hasService", "owner" and "administrator" shown in FIG. 1 and their inverses).

Name	**usesControlledList**
Term URI	http://iesr.ac.uk/terms/#usesControlledList
Label	Uses Controlled List
Defined By	http://iesr.ac.uk/terms/#usesControlledList
Source Definition	A classification scheme or thesaurus used by the collection
Has Encoding Scheme	http://iesr.ac.uk/terms/#CtrldVocabsList
Data Type	<string>
Occurrence	Min: 0; Max: unbounded
IESR Searchable	20, 1040, 1112, 1016, 1017, 1035; classn, anywhere

FIG. 2. IESR Application Profile entry for a Collection property.

3.1. Collection Properties: Importing and Customising an Existing Application Profile

As explained above, the IESR collection metadata is based on the RSLP Collection Description Schema (RSLPCD), which at the time when the IESR project started appeared to be an appropriate schema to use for collection description, even though it was a 'de facto' rather than 'official' standard. The DCMI Collection Description Application Profile (DCMI, 2007a) and the NISO Metasearch Initiative Collection Description Specification (ANSI/NISO, 2005) have since been derived from RSLPCD, so IESR has tried to maintain consistency with these schemas and has fed experience into their development.

RSLPCD was developed to describe both physical and electronic collections within a range of domains, including museums, archives and libraries. Because IESR describes electronic collections, primarily to support discovery, not all of the RSLPCD properties have been included in IESR. For example records management information such as details about accrual, legal and custodial history were not thought relevant within IESR. In the early stages of the IESR project potential stakeholders were surveyed about their possible use and requirements of a registry of collections and services. This provided initial input to decisions on the properties that would be needed or useful. These decisions have been refined over time as use has been made of IESR with 'formal' metadata reviews undertaken at intervals. In some cases the IESR definition of a term is a refinement of the base RSLPCD or Dublin Core definition, and there are additional IESR comments, all of which are stated in the Application Profile.

IESR also identified some further properties that were needed within the domain, which is the JISC Information Environment, or the application. For example, "usesControlledList" captures the subject vocabulary that a collection uses to describe its items. This term was introduced at the request of a proposed terminology service (Nicholson et al., 2006), but it will also be useful to a portal that wishes to provide to its users an item level search over collections discovered in IESR. A second example is the "hasService" property that relates a collection to a service.

3.2. Collection Properties: Adding New Terms to an Application Profile

Although it was IESR's intention to use standard metadata terms, it seems generally better to introduce a new term when there does not appear to be a suitable existing one. The alternative would be to 'shoehorn' a new definition into an existing term, or to adopt a term from an obscure namespace. A certain amount of care is needed if selecting a term from another namespace. Not only does its definition have to be appropriate, it also has to be a property-value assertion as Dublin Core terms are. A term that is an element within a hierarchical XML data model is not a suitable candidate for inclusion in a DCAP because it can exist only within that hierarchy, with no stand-alone meaning.

Terms within an Application Profile are identified by URIs. Many of the properties in RSLPCD are in fact Dublin Core properties so have persistent URIs. The RSLPCD properties also have URIs assigned. For new IESR terms, URIs have been assigned within an IESR namespace with an intention of persistency. Some terms, e.g. "itemType", are now available within the DCMI Collection Description Terms namespace (DCMI, 2007b). But making changes becomes problematic within an existing application, especially when developments of a standard are still fluid. So such a term will remain within the IESR namespace, but a comment within the Application Profile indicates its mapping to the corresponding standard term. The same considerations apply to vocabulary encoding schemes for property values, several being defined within the IESR namespace. Currently each IESR term is defined in a human readable 'mini application profile' with its URI grounded on its position in that document. But ideally they should eventually be defined by machine readable RDF assertions.

3.3. Service Properties: a Bespoke Application Profile

IESR Service description is a bespoke schema because there did not appear to be a suitable existing scheme to describe a wide range of service types. Other requirements for service description were extension to include IESR terms, such as the inter-entity relations, and consistency with the 'flat' Dublin Core data model, rather than a hierarchical XML model. Specific service connection details are captured externally to IESR, via an "interface" property whose value is a URI that references machine readable information according to the appropriate standard for the service protocol, e.g. Web Services Description Language (WSDL) (Christensen, 2001). Because IESR is agnostic about service protocol it differs from a UDDI-based registry, which would register only WSDL described Web Services.

The advantage of a bespoke service description schema defined by an Application Profile has been the ability to extend it by adding new properties, as further requirements have been identified. For example, properties have been added to support the capture of information related to a particular authentication model. And during the latest metadata review a property was added to enable the description of an OpenURL resolver including its preferred link text in addition to an image button.

Several vocabulary encoding schemes are defined within the IESR namespace for service property values. These include a list of available service protocols (or access methods), and an indication of the protocol compliance level or version supported. A recent inclusion is a list of service functions (or genres). This is based on a possible service function list for ISO 2146 (Pearce & Gatenby, 2005), with some additions from the e-Framework (e-Framework, 2007) service genre list and others identified by IESR. This vocabulary may be refined as its use becomes apparent in practice, an option made possible by its definition within the IESR namespace.

4. From Application Profile to XML Serialisation

IESR disseminates data descriptions in XML, apart from its web interface, which displays equivalent text fields. Similar XML data is also used internally within the application. Records disseminated via Z39.50 are composite description sets containing a collection entity, all its associated services and agents, and administrative metadata components for each entity. Records disseminated via the OpenURL 'link to' resolver interface, or by simply dereferencing a URI, are single entities, again a description set containing the entity and its associated administrative metadata. From the OAI-PMH interface the single entity metadata record is the sole component of the description set, the administrative metadata being within a separate 'about' section as prescribed by the protocol.

The IESR XML schema (IESR, 2007c) is based on the Application Profile. The elements within an XML record for an entity correspond to the properties defined in the Application Profile. XML attributes accord with vocabulary encoding schemes specified in the Application Profile. However it was not possible to create an XML schema that is strictly compliant with the Application Profile, because of limitations of the XML schema specification and because of some imported standard schemas.

The IESR XML schema does not impose occurrence requirements such as mandatory or non-repeatable elements. It did not seem possible to define an XML schema that allowed elements to be in any order as well as specifying occurrence. Freedom in ordering was thought to be more important. Thus within the IESR application, when records are added to the registry, data validation checks are made in addition to verifying the XML by parsing.

Many properties within the IESR schema are from Dublin Core, so the XML schema imports the Simple and Qualified Dublin Core XML schemas (DCMI, 2006). The definition of those schemas means that element refinements are implicit. So an XML schema cannot explicitly reference both an element and its refinement, such as 'dc:title' and 'dcterms:alternative'. Thus such element refinements are included within comments in the IESR XML schema.

Another consequence of importing the original Dublin Core XML schemas, which were current when the IESR XML schema was defined, is that vocabulary encoding schemes cannot be constrained by the XML schema. Dublin Core XML uses the 'xsi:type' attribute to indicate the appropriate encoding scheme for a property. Because this attribute is a pre-defined part of the XML language it is not possible to constrain its value. Thus vocabulary encoding schemes are also indicated within the XML schema by comments and checked by data validation on registration.

It is anticipated that new guidelines about encoding Dublin Core in XML will be available soon (Johnston & Powell, 2006a), superseding the version used for IESR. These guidelines will allow a vocabulary encoding scheme constraint. However it is unlikely that IESR will change its XML schema to conform to these new guidelines because it would imply too radical a change to an XML encoding that is already in use. However, IESR will create a DC-Text (Johnston & Powell, 2006b) example to illustrate a mapping to the new DC-in-XML guidelines and conformance to the Dublin Core Abstract Model.

5. Using the Application Profile: a General Service Registry Application Profile

There has been increasing interest in use of the IESR metadata schema by other similar initiatives. The documentation of the metadata schema using a DCAP is invaluable in assisting such use. In the USA the OCKHAM Digital Library Service Registry (Frumkin, 2004) is using IESR metadata to describe its services and collections, as is the registry that is part of the aDORe Digital Object Repository (Van de Sompel et al., 2005) at Los Alamos National Laboratory Research Library. The Australian Partnership for Sustainable Repositories (APSR, 2007) intends to use the IESR metadata schema for their collection registry project. It has become apparent that

the constraints defined in the IESR Application Profile specifically for the IESR application may be unsuitable for other uses. The occurrence constraints may be too strict and the vocabulary encoding schemes may be inappropriate. In particular the aDORe repository, being a machine-based application, uses a cut-down selection of properties from the IESR Application Profile, with no current scenario requiring an agent description.

Thus a laxer, more general version of the IESR Application Profile has been produced. The IESR DCAP document contains both the general Application Profile and the stricter IESR Application Profile. IESR constraints above the general ones are introduced with further IESR-specific application profile attributes: 'IESR Occurrence', 'IESR Condition', 'IESR Data Type' and 'IESR Has Encoding Scheme'.

It is possible that other users of the IESR schema may wish to extend the metadata to capture further information, yet remaining compliant to the IESR Application Profile and hence interoperable with IESR. An additional 'more information' property ('seeAlso') has been added to the IESR Application Profile for all three entities, whose value is a URI. Potentially this could point to further machine-readable XML providing extra information. Currently this property is not used by IESR itself, but may appear in data harvested from elsewhere. Possible uses of 'more information' may be to indicate official endorsement of a collection, or to capture institution profile details for an agent.

6. IESR Use Cases

The specification of the IESR Application Profile and the development of the application has largely been a theoretical activity, with only a modicum of experimental use. The purpose of IESR is still viewed as visionary, with little general understanding of the use of a middleware service registry. The JISC Information Environment is seen by many as simply serving resources to people through web pages. In order to suggest ways in which IESR could be employed a set of hypothetical use cases with associated scenarios has been developed (Apps, 2007a).

The over-arching use expected of a service registry is shown in FIG. 3. A 'Service' (and collection) is 'registered', by a Contributor who is probably the service administrator. A 'Client', such as a portal application, 'discovers' a relevant service within the 'Registry'. It 'invokes' the service by locating it using the details ascertained from the Registry, and then connecting to it possibly using further interface details acquired from the Registry.

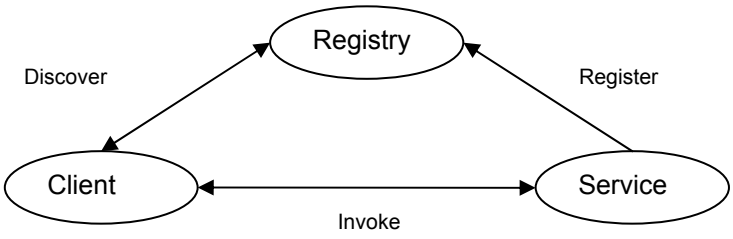

FIG. 3. Using a service registry.

6.1. Use Scenarios

The primary use case for IESR is dynamic discovery by an application such as a portal, the 'client' of FIG. 3, to support the functionality it provides to its end users. Thus discovery is likely to be subject based relevant to a user's discipline. This use case could fulfil scenarios such as Mary, a physicist, using a metasearch portal to perform a literature search about Higgs-Boson particles, Mike, a medical researcher using a medical portal to find information about treatments for Alzheimer's disease, Paul using his institutional portal to look for resources about

tuberculosis transmission, or Sam using a union catalogue to find a book about microlight aircraft. Similar use scenarios could be supported by a client application that maintains its own local registry by harvesting IESR data, possibly merging it with data from other sources. An application may wish to maintain a local registry for performance reasons, to obviate the need for repeated network accesses to IESR. Or an application such as a metasearch portal may wish to transform IESR data into a proprietary format to populate a local knowledgebase.

A secondary use case is discovery of information in IESR by a person. A builder of a portal application may think that using IESR dynamically is too high a barrier for current development. So they could discover details of appropriate collections and services in IESR and use that information to plug relevant services into their portal. For some service protocols, in particular SOAP Web Services, only this static use of IESR is possible, the semantics of the interfaces of such services being proprietary. Use scenarios based on this use case include Jane who wants to plug a bibliographic service into an eResearch bioinformatics Grid portal, David who is building an OpenURL resolver, and Kate, an aeronautical engineer, who is setting up her personal RSS (Wikipedia, 2007b) portal.

Or IESR could be used by a person simply to search for relevant resources. Use scenarios here include Grace, a physics librarian, creating an annotated list of appropriate resources, Margaret, a materials science lecturer, finding suitable resources to recommend to her students, Stephen, a historian, discovering resources via an IESR search box within his institutional portal, and researchers, Colin and Jackie, looking for OpenURL resolvers, either their institution's or freely available.

Other service applications could use IESR dynamically. Use scenarios include services that check the availability or locator addresses of Z39.50 services, and a repository's format validation option.

6.2. Interaction Between IESR Use Cases and the Application Profile

The Application Profile proved invaluable during development of the use cases. It effectively lists a set of properties each of which would be expected to have a purpose. It is fairly obvious that discovery will use such properties as 'title', 'abstract' and 'subject'. But an exhaustive set of use cases ideally should include a use for all properties, especially those introduced by IESR, such as 'usesControlledList', but also those introduced as part of a 'base' schema, which is RSLPCD in IESR's case.

Conversely, analysis of the use cases informed additions to the Application Profile, by indicating necessary details currently missing from the IESR metadata, both properties and vocabulary terms. For example, use scenarios about research literature searches imply an additional collection item type to indicate a 'scholarly work'. The list of service functions has been completely revised recently and can now support use scenarios such as the above example concerning format validation. A use scenario about using IESR records to part-populate a local OpenURL resolver registry indicated the need for a property to capture a resolver's preferred link text. Several use scenarios suggested the inclusion of a 'status' property within the administrative metadata to indicate the currency and quality of a record, or whether it has been harvested.

Hopefully the use scenarios will suggest actual, demonstrable use of IESR. Real testing of IESR should further refine the list of properties within the Application Profile. There are currently plans to develop actual use scenarios within a JISC Information Environment 'test bed' project, which will include use of IESR. Concurrently the OCKHAM Digital Library Service Registry is conducting a Registry Experiment. Data records will be shared by harvesting between the OCKHAM registry and IESR. It is the intention to demonstrate registry use by several identified projects funded by the US National Science Digital Library. It is possible that such testing will indicate omissions in the IESR metadata, and it will be instructive to observe whether some properties are not beneficial.

7. Data Creation

7.1. Manual Data Creation

IESR provides a data Editor, a web form based application, for data contributors to supply descriptions of their collections, services and agents. Within the Editor there is a separate input form for each entity. The entity input form effectively replicates the properties listed in the Application Profile.

Several issues have arisen from this method of data input, mainly related to how onerous the task is. In particular the list of collection properties is long, which reinforces concerns about the necessity of some of them. Attempts have been made to improve the input form by reordering so that the required properties appear first. A further change suggested is to move some of the optional properties to a second 'page'. A recent study of the properties in relation to the use cases has proposed a set of 'recommended' properties in the hope of simplifying data input.

A criticism of IESR data input is that creating service descriptions requires a certain level of technical knowledge. However for service descriptions to be used by other applications these technical details are necessary. Some service properties are conditional on the value of others. For example, 'interface' is relevant to only some service protocols, 'domain available' is only relevant for institution OpenURL resolvers and library catalogues, and some properties are applicable only to services that use a particular authentication method. Therefore the Editor shows only relevant properties, in an attempt to improve usability.

A significant issue in data creation is declaration of the relations between the entities. It seems difficult to explain the concept of and necessity for this inter-entity linking to contributors. A future exercise will be to consider the usability of this application and explore whether this aspect can be made easier. It has become clear that data creation according to an Application Profile that specifies metadata properties for a single entity is much easier to comprehend. But the multiple, related entities of IESR are an essential consequence of the domain model.

7.2. Data Harvesting

An alternative method of data creation is data harvesting from other registries, or from contributors who wish to batch load large numbers of records. IESR provides OAI-PMH harvesting of collection, service and agent descriptions that conform to the IESR XML schema and Application Profile. This should provide a relatively painless way to supply records to IESR, although contributors will still need to cope with some of the above considerations. The use of OAI-PMH will enable automatic updating of changed records.

But some registries that have been suggested as possible sources of IESR data descriptions do not supply data in IESR XML format, and some have a proprietary API (Application Program Interface) rather than a standard OAI-PMH interface. Thus setting up ingest of data from these registries will necessitate specific software routines on a case-by-case basis.

A further issue is that the data in registries identified for possible harvesting is not rich enough for inclusion in IESR and may require some augmentation. In all cases where an application suggested for harvest does not supply IESR XML, a data mapping will be needed. This will consist initially of comparing the potential supply data with the IESR Application Profile, illustrating a further important use for an Application Profile.

8. Conclusion

The experience of using an Application Profile in the IESR project has shown it to be invaluable for formally documenting a metadata schema. It provides a document for discussion during initial development of the schema, and for communicating the metadata schema to stakeholders, who may be potential users of the metadata schema or the application. The Application Profile provides a clear specification even to those who are not conversant with

metadata schemas. It affords a relatively 'syntax free' format understandable by non-technical people. The IESR Application Profile is a web document, so it includes hyperlinks between various sections and definitions, which hopefully enhance usability by readers. At the same time it is regarded as a formal specification with a persistent URI.

Further experience with employing an Application Profile, within a project outside of the digital library domain, has reinforced the belief in its importance as a basis for discussion and communication about a metadata schema. Part of the JISC Information Governance Gateway (JIGG) project involves developing a repository of Freedom of Information (FOI) disclosure log entries that summarise FOI requests received by UK Higher Education Institutions and the information released (Apps, 2007b). As well as being searchable the repository will share and gather records via OAI-PMH. To support this application a set of fields within an FOI disclosure log has been defined, and documented in an Application Profile. The Application Profile proved to be an ideal format to assemble, communicate and discuss suitable properties during the process of gaining agreement, and for dissemination of the details to other interested parties. This was within a sector where there was not general awareness of metadata schemas and no previous knowledge of OAI-PMH.

The Application Profile provides a theoretical specification for an application, translating the domain model into concrete data semantics. This paper has shown how the application is built around the Application Profile. But the Application Profile is not a static document. It has been refined during the on-going experience of developing the project into service. This obviously raises versioning issues. IESR maintains a persistent URI to the latest version of the Application Profile, actual versions being indicated by their creation date within their URIs. Similar dated versioning applies to the XML schema, but generally the dated URI is used where XML verification will occur, for example within OAI-PMH records. Once the initial Application Profile was defined, changes have been managed by formal metadata reviews at spaced intervals, continual changes seeming ill advised. As IESR has matured, consideration has been given to backwards compatibility, changes that impact on existing XML data being made only where absolutely necessary.

A concern that is emerging as IESR matures is the quantity of properties specified in the Application Profile for each entity. There seems to be a danger, especially when importing an existing Application Profile, of including an ambitious number of properties that have a perceived theoretical use. This problem can be compounded if stakeholder suggested properties are also introduced. A large number of properties discourages data contribution as well as adding to the burden of data management. Experience now suggests that some pragmatic decisions should be made to pare down the set of properties in an Application Profile to those that have a practical use.

The Application Profile has proved to be useful not just for documenting the metadata schema but also for informing the XML schema, application development and possible use scenarios, work which fed back into refinement of the Application Profile. Having an Application Profile has made it easier to share the IESR metadata schema with others, initiatives that have led to a generalisation of the Application Profile.

Thus experience with developing the IESR indicates that an Application Profile can provide the central data specification for an application's development and promotion. It is at the heart of realising the domain model into a concrete application disseminating useful data.

Acknowledgements

IESR is a 'service in development' project funded by the Joint Information Systems Committee (JISC) of the UK Higher and Further Education Funding Councils as part of its 'shared infrastructure services programme'.

References

ANSI/NISO. (1995). *Z39.50-1995 Informational Retrieval*. Retrieved March 16, 2007, from http://www.niso.org/standards/resources/Z39-50.pdf.

ANSI/NISO. (2005). *Z39.91-200x Collection Description Specification*. Retrieved March 16, 2007, from http://www.niso.org/standards/resources/Z39-91-DSFTU.pdf.

Apps, Ann, and Ross MacIntyre. (2006, May). Why OpenURL? *D-Lib Magazine, 12(5)*. Retrieved March 16, 2007, from http://www.dlib.org/dlib/may06/apps/05apps.html.

Apps, Ann. (2004). A registry of collections and their services: From metadata to implementation. *Proceedings of the International Conference on Dublin Core and Metadata Applications, Shanghai, China* (pp. 67-73).

Apps, Ann. (2006). Disseminating service registry records. *Proceedings of the Tenth International Conference on Electronic Publishing, Bansko, Bulgaria* (pp. 37-47).

Apps, Ann. (2007a). *IESR use cases*. Retrieved March 16, 2007, from http://iesr.ac.uk/use/use-cases/.

Apps, Ann. (2007b). Disclosing freedom of information releases. *Proceedings of the Eleventh International Conference on Electronic Publishing, Vienna, Austria* (pp. 425-434).

APSR. (2007). *Australian Partnership for Sustainable Repositories: APSR Projects*. Retrieved March 16, 2007, from http://www.apsr.edu.au/currentprojects/index.htm.

CEN. (2003). *CWA 14855: Dublin Core Application Profile Guidelines*. Retrieved March 16, 2007, from ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/MMI-DC/cwa14855-00-2003-Nov.pdf.

Christensen, Erik, Francisco Curbera, Greg Meredith and Sanjiva Weerawarana. (2001). *Web Services Description Language (WSDL) 1.1*. Retrieved March 16, 2007, from http://www.w3.org/TR/wsdl.

DCMI. (2006). *XML schemas to support the Guidelines for Implementing DC in XML recommendation*. Retrieved March 16, 2007, from http://www.dublincore.org/schemas/xmls/.

DCMI. (2007a). *Dublin Core Collection Description Application Profile*. Retrieved March 16, 2007, from http://www.dublincore.org/groups/collections/collection-application-profile/.

DCMI. (2007b). *Dublin Core Collection Description Terms*. Retrieved March 16, 2007, from http://www.dublincore.org/groups/collections/collection-terms/.

e-Framework. (2007). *Service Genre Registry*. Retrieved March 16, 2007, from http://www.e-framework.org/Services/Genres/ServiceGenreRegistry/tabid/655/Default.aspx.

Frumkin, Jeremy. (2004). The problem of mainstreaming digital libraries. *OCLC Systems and Services: International Digital Library Perspectives, 20(3) pp* 106-109.

Heaney, Michael. (2000). *An analytical model of collections and their catalogues*. Retrieved March 16, 2007, from http://www.ukoln.ac.uk/metadata/rslp/model/amcc-v31.pdf.

IESR. (2007a). *JISC Information Environment Service Registry*. Retrieved March 16, 2007, from http://iesr.ac.uk/.

IESR. (2007b). *IESR Application Profile*. Retrieved March 16, 2007, from http://iesr.ac.uk/profile/.

IESR. (2007c). *IESR XML Schema*. Retrieved March 16, 2007, from http://iesr.ac.uk/schemas/xsd/iesr.xsd.

JISC. (2007). *JISC information environment*. Retrieved March 16, 2007, from http://www.jisc.ac.uk/whatwedo/themes/information_environment.aspx.

Johnston, Pete, and Andy Powell. (2006)(a). *Expressing Dublin Core metadata using XML*. Retrieved March 16, 2007, from http://dublincore.org/documents/dc-xml/.

Johnston, Pete, and Andy Powell. (2006)(b). DC-Text: A simple text-based format for DC metadata. *Proceedings of the International Conference on Dublin Core and Metadata Applications, Mexico* (pp. 24-30).

Lagoze, Carl, Herbert Van de Sompel, Michael Nelson, and Simeon Warner. (2004). *The Open Archives Initiative Protocol for Metadata Harvesting*. Retrieved March 16, 2007, from http://www.openarchives.org/OAI/openarchivesprotocol.html.

Nicholson, Dennis, Alan Dawson, and Ali Shiri. (2006, May). HILT: A terminology mapping service with a DDC spine. *Cataloging and Classification Quarterly, 42*(3/4), 187-200. Retrieved March 16, 2007, from http://eprints.rclis.org/archive/00008767/.

Pearce, Judith, and Janifer Gatenby. (2005). *New frameworks for resource discovery and delivery*. Retrieved March 16, 2007, from http://www.nla.gov.au/nla/staffpaper/2005/pearce1.html.

Powell, Andy, Michael Heaney, and Lorcan Dempsey. (2000, September). RSLP Collection Description. *D-Lib Magazine, 6*(9). Retrieved March 16, 2007, from http://www.dlib.org/dlib/september00/powell/09powell.html.

Powell, Andy, Mikael Nilsson, Ambjorn Naeve, and Pete Johnston. (2005). *DCMI Abstract Model*. Retrieved March 16, 2007, from http://www.dublincore.org/documents/abstract-model/.

Van de Sompel, Herbert, Jeroen Bekaert, Xiaoming Liu, Luda Balakireva, and Thorsten Schwander. (2005). aDORe: A Modular, Standards-Based Digital Object Repository. *The Computer Journal, 48*(5), 514-535.

Wikipedia contributors. (2007a). Service-oriented architecture. In *Wikipedia, The Free Encyclopedia*. Retrieved March 16, 2007, from http://en.wikipedia.org/w/index.php?title=Service-oriented_architecture&oldid=115525599.

Wikipedia contributors. (2007b). RSS. In *Wikipedia, the tree encyclopedia*. Retrieved March 16, 2007, from http://en.wikipedia.org/w/index.php?title=RSS&oldid=115357482.