Development of a Native Cross-Platform iRODS GUI Client

Ilari Korhonen, Miika Nurminen
IT Services, University of Jyväskylä
PO Box 35, 40014 University of Jyväskylä, Finland
ilari.korhonen@jyu.fi, miika.nurminen@jyu.fi

ABSTRACT

This paper describes activities on the research IT infrastructure development project at the University of Jyväskylä. The main contribution is a cross-platform iRODS client application with a rich graphical user interface. The client application is fully native and builds from a single C++ codebase on all of the platforms on which iRODS 4.0 is supported. The application has a responsive UI with native look & feel and enables drag & drop integration to the desktop. This is made possible by basing the development of the client application on top of the Qt 5 framework and an object-oriented C++ framework for iRODS which is being developed with the client application. The object-oriented framework wraps around the native iRODS 4.0 C/C++ client API library and provides object-oriented interfaces to iRODS protocol operations e.g. a fully object-oriented iRODS General Query (GenQuery) interface used by the client application has been implemented in this C++ framework. By developing on top of the native C/C++ iRODS API library, the plugin architecture of iRODS 4.0 can be fully leveraged in authentication (e.g. Kerberos) and network transport (e.g. SSL) modules without any additional complexity.

Keywords

Research data, metadata management, infrastructure, iRODS, client software, graphical user interface.

INTRODUCTION

There is an increasing demand for IT services for researchers that span the full ”stack” of storage and computation infrastructure for research data with metadata support and widely available interfaces for information extraction and reporting. On one hand, the scope of computationally intensive datasets currently spans all fields of research necessitating university-wide support for scientific computing and research data management. On the other hand, funders and institutions (e.g. EU’s Horizon 2020) have expressed an increased demand for opening all research materials related to publicly funded research. This can be seen as a continuation of recent development with institutional repositories (e.g. DSpace, EPrints, and Fedora) supporting the ”green” way of Open Access for publications.

In Finland, the National Research Data Initiative (TTA) and Open Science and Research (ATT) projects have developed research data infrastructure and promoted open access. The Ministry of Education and Culture is considering to include openness (at least for publications) as an element in the funding model for the Finnish universities [1], pushing the need to get (meta)data from research data as well. In this paper, we describe research IT infrastructure development at the University of Jyväskylä, focusing on iRODS. An iRODS client application is introduced and briefly evaluated with respect to planned data management processes. Finally, prospects for development are outlined.

RESEARCH IT INFRASTRUCTURE DEVELOPMENT AT THE UNIVERSITY OF JYVÄSKYLÄ

In this paper, we describe some of the recent development efforts related to research IT infrastructure at the University of Jyväskylä. The discussion is focused on the project codenamed ”Kanki” (=meaning e.g. in Finnish ”a rod” and in Japanese ”cold” or ”frost”) – a native cross-platform iRODS client application with a rich graphical user interface

iRODS UGM 2015 June 10-11, 2015, Chapel Hill, NC

based on Qt[1] framework. The client application is targeted towards researchers of various disciplines as well as other interest groups utilizing or curating research data (e.g. librarians), possibly lacking the expertise to use the iRODS command-line interface. Our client application will enable the users to utilize the full power of an iRODS data grid complete with powerful data management functions such as schema management and validation of metadata.

Background

There is a relatively long tradition related to the advancement of open access and research data management at the University of Jyväskylä. Theses were published online by the Jyväskylä University Library as early as 1997, leading to the introduction of the DSpace-based institutional repository JYX[2] in 2008 [2]. A university-wide working group for parallel publishing and administration of research material was commenced in 2009, resulting in the development of a mechanism for parallel publishing of publication files from the research information system TUTKA[3] to the institutional repository. Even though the working group had identified various types of research materials that should be preserved, support for managing research data in a standardized way (considering both tools and data-related processes) was incoherent, differing between research groups. It proved to be of considerable difficulty to advance standard data management practices when the research itself is done independently of administrative processes, often using specialized tools and software for e.g. analysis or other parallel computation on datasets. Many older research materials are still in analogue form – and even those in digital form are often stored in either removable media, portable hard drives or in the best case – file servers. Some faculty-specific solutions such as YouData[4] are in use, but most datasets lack standardized metadata descriptions, complicating data discovery and reuse.

Recently, University of Jyväskylä has taken an active approach on managing research data and infrastructures. In September 2014, JYU was the first Finnish university to have published its official principles for research data management [3]. The development project for research IT infrastructure and research data management has been active in 2013-2015. Project activities include the adoption and integration of the Dataverse Network[5], the development of a university-wide iRODS grid infrastructure for research data storage, and the surveying of essential datasets in the faculties for which to develop iRODS data management services. The iRODS platform has been selected as the primary focus of development activities in the project. Overall architecture is based on separated responsibilities between the systems. Even though some institutional repositories have been augmented for publishing research data, support for managing the data during the whole research life cycle is inadequate (unless extensively customized, which can be a problem from maintenance point of view). The metadata used in research datasets has considerably more variation compared to metadata typically used in the repositories (e.g. Dublin Core). Providing access to data in repositories is no longer enough since people want to *do things* with that data [4]. iRODS responds to this need internally. Dataverse has potential to respond to the external needs with citable datasets and analysis functionality.

The iRODS-related development activities at JYU include the development of a secure, scalable, high-performance, high-availability iRODS data grid infrastructure for university-wide deployment with infastructure automation, the development of server-side iRODS modules for e.g. metadata autoextraction and data anonymization, and finally, the development of a native cross-platform iRODS GUI client to enable schema-based metadata management with validation capabilities and to serve as a platform for future iRODS-based applications. A common metadata model for JYU research data management is being developed with JYU Library, based on national specifications. The Finnish national research data storage service IDA[6] – maintained by the Finnish IT Center for Science CSC – is built on top of iRODS as well. Collaboration with the IDA development has been planned. The overall goal of the project is in the advancement of the IT service culture to improve the acceptance of centralized services among the researchers and to be able to provide added value compared to isolated legacy solutions.

---

[1] http://www.qt.io/
[2] https://jyx.jyu.fi/
[3] http://tutka.jyu.fi/
[4] http://youdata.it.jyu.fi/
[5] https://dvn.jyu.fi/
[6] https://www.tdata.fi/en/ida

Infrastructure Development

The IT infrastructure at the University of Jyväskylä is a largely consolidated one with most of its servers residing as virtual machines in a VMware vSphere 5 cluster. Separate physical servers are being used alongside virtualization for performance critical computing or I/O-intensive applications while strongly favouring virtual servers. Shared storage is being provided by EMC VNX Series SAN/NAS unified storage arrays connected to Fibre Channel and 10 Gbps Converged Enhanced Ethernet fabrics. After initial evaluation and testing of iRODS it was promptly concluded that a single deployment of an iRODS iCAT server is insufficient to provide the performance targeted for scalable use and would not be highly available without the use of (performance limiting) hardware virtualization. This prompted the design of a scalable and inherently highly available infrastructure for iRODS iCAT deployment at JYU.

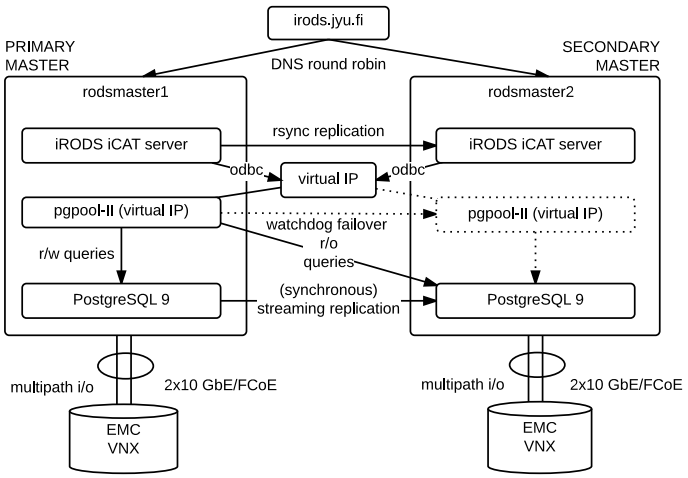

Figure 1. Illustration of the scalable HA model for JYU iRODS deployment.

A critical point from performance point of view and simultaneously a *single point of failure* is the database server used to host the iRODS iCAT database. For the database instance to be able to both scale out from a single server, and to withstand a loss of a server without compromising the integrity of the database – a properly clustered database solution is a necessity. An Oracle RAC database high-availability cluster solution wasn't considered feasible because of the prohibitive pricing per processor core of Oracle database server products. After evaluating the possible alternatives (e.g. HAIRS [5]) for a high-availability load balanced iRODS iCAT cluster we propose the following model for a scalable iRODS iCAT deployment (Figure 1). The solution is built on a highly available pooled configuration of a PostgreSQL 9 database, using PostgreSQL 9 streaming replication and PgPool-II for load balancing and HA failover on top. The system can be built on configurations with at least two servers and scales horizontally on read-only transaction performance and is able to withstand the loss of a single server and having no single point of failure.

On the very lowest level of the configuration are the PostgreSQL 9 database server instances, one initially set up as the primary database master, the other ones as read-only database replicas, which receive streaming replication from the master database. On top of the PostgreSQL 9 database server instances is PgPool-II, which is configured to a HA configuration via the built-in watchdog facility. PgPool-II is configured on both or all of the iCAT servers to *share* an IP address (in an HA subnet available in a private VLAN shown to the iRODS iCAT servers) such that if the current master PgPool-II host goes down there will be an *escalation* procedure to select a new PgPool-II master host which will take over the *virtual* PostgreSQL pool IP address in the HA subnet. The high-availability PgPool-II configuration – resident in all of the iRODS iCAT servers connected via the HA subnet – is aware of the state and health of all of the underying database servers and in the case of a failover event (master PostgreSQL database backend health check failed) executes a *recovery* operation to an available PostgreSQL 9 hot standby server, which becomes the new master of the database as ordered by PgPool-II and starts accepting read-write transactions to the database. The recovery process sets the other server(s) as read-only hot standby replicas of the new master database.

The iRODS iCAT server(s) are set up on top of the PgPool-II managed database cluster such that the iCAT server(s) are set to use a pooled database available at the pool virtual IP. This way the iRODS iCAT may utilize the entire database cluster for increased performance, since the PgPool-II middleware load balances the read-only transactions throughout the cluster. Read-write transactions are sent only to the master database and synced to the other nodes with streaming replication. Some load balancing of the iRODS connections can be accomplished using a simple DNS round-robin set up with the caveat that iRODS itself is not DNS round-robin aware. Additionally the DNS name service caching in use in common operating systems hinders with the round-robin of resolved IP addresses. We hope that in the future this aspect would be addressed in the development of iRODS. For the time being this solution provides some load balancing between clients of iRODS connections to two or more iRODS iCAT servers.

The deployment of new iRODS instances to a HA pair of servers is being done at JYU with Ansible infrastructure automation. The setup has been parametrized such that a new iRODS instance can be specified simply by Ansible *group variables* for the group of iRODS iCAT servers and *host variables* to specify the hosts themselves. This is believed to be useful not only for quick deployment of development servers but also for deploying new iRODS instances for special use cases. For example, a specialized instance of iRODS deployment is in planning stage for the new Jyväskylä Center for Interdisciplinary Brain Research[7].

PROJECT KANKI – A NATIVE CROSS-PLATFORM GUI CLIENT APPLICATION FOR IRODS

During the research IT infrastructure development project several needs have risen for iRODS-based research data (and *meta*data) management. These prompted the need for iRODS user interface development. The utmost important of these was the need for secure data and metadata transfer. Other specific needs not properly accomodated by other existing freely available solutions included the graphical iRODS search tool with arbitrary search criteria formation for data discovery, metadata schema management with visual namespace and attribute views and readiness for metadata schema validation for data quality assurance. Some open source projects provided these features partially, but other projects such as Davis[8] were discontinued and thus rendered unsupported after the introduction of iRODS 4.0.

About the Development

To implement these user interface features for iRODS-based research data management at JYU, a software project was started – eventually codenamed ”Kanki” – to build an iRODS 4.0 compatible client application with integration to Kerberos authentication, the option to use iRODS 4.0 SSL secured connections, to develop extensible data and metadata management features, and to serve as a framework for iRODS integration to different kinds of scientific software. The introduction of iRODS 4.0 and the incorporation of the modular architecture in both iRODS 4.0 server and client side made the native iRODS client library more attractive for client-side development than any of the other options available. The possibility to use e.g. Kerberos authentication or SSL transports in the iRODS connections out-of-the-box – without having to resort to e.g. IPSec for transport layer security – made a convincing case for C++ to be used also for client-side development instead of more popular alternatives like Java or Python.

Since the goal of the development was to produce a cross-platform application while still remaining fully native, the widely adopted Qt framework proved to be an excellent choice as a development platform. At the University of Jyväskylä we deploy all of the major platforms i.e. Linux, Mac OS X and Windows, with Red Hat Enterprise Linux being the prominent Linux distribution with a campus license. For developing a cross-platform application targeted to all these platforms, the Qt framework provides exceptional support for compiling from a single codebase.

Challenges

A working build configuration for Mac OS X took some effort since the newer versions of OS X, namely versions later than OS X 10.8 Lion caused some difficulties for building iRODS. The version of the boost libraries compatible with iRODS 4.0 proved to be incompatible with the OS X provided clang compiler. GCC 4.8 or later proved to be a

[7] http://cibr.jyu.fi/
[8] https://code.google.com/p/webdavis/

working solution. There were other issues as well caused by a dynamic linker symbol conflict with OS X bundled MD5 library functions and ones provided with iRODS. The most severe consequence of this seemed to be the inability to use the native auth module in OS X builds of iRODS 4.0, since the runtime dynamic linker resolved some of the MD5 symbols to the iRODS bundled ones and others to the OS X provided ones, causing the corruption of the memory buffer used to compute the MD5 challenge response. This was first worked around by not using native authentication – which is not to be used at JYU iRODS at all anyway. A workaround solution was found for the issue by changing linkage of the auth module in a way which resolves the symbol conflict. Currently, this issue has been resolved in the iRODS master branch.

Windows still remains as an unsupported platform since iRODS 4.0 isn't Windows compatible at the time of writing of this paper. With Windows support added to the iRODS codebase our client can be built on Windows as well.

Features

The client (see Figure 2) is intended to eventually serve as a bona fide alternative user interface to iRODS icommands – the reference user interface for iRODS. Implementing all of this functionality in a native "desktop" application will enable the users to harness the full power of iRODS with native application performance and the usability of a graphical user interface.

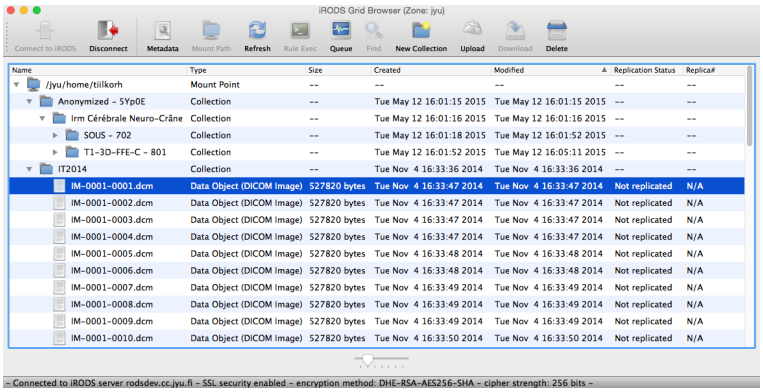

Figure 2. iRODS Grid Browser window in the client application.

Additionally, the client has some specialized features for metadata management not found in currently available iRODS clients. Metadata schema management has been implemented with features like namespace and attribute management. Below is an example of attribute descriptions in the XML metadata schema configuration.

```
<irods:namespace prefix="fi.jyu.irods." label="University of Jyväskylä">
  <irods:attribute name="metadata.modified" unit="false" editable="false">
    <irods:label>Metadata Modification Time</irods:label>
    <irods:displayFilter type="regExp"><irods:regExpRule>(\d+)-(\d+)-(\d+).(\d+):(\d+):(\d+)</irods:regExpRule>
    <irods:regExpFilter>\3.\2.\1 \4:\5:\6</irods:regExpFilter></irods:displayFilter>
  </irods:attribute>
  <irods:attribute name="language" unit="false" editable="true">
    <irods:label>Language</irods:label><irods:values strict="true">
    <irods:value name="ISO6392:FIN"><irods:label>Finnish</irods:label></irods:value>
    <irods:value name="ISO6392:ENG"><irods:label>English</irods:label></irods:value>
    <irods:defaultValue>ISO6392:ENG</irods:defaultValue>
  </irods:values></irods:attribute>
</irods:namespace>
```

In the metadata schema configuration namespaces are identified along with attributes defined in the namespaces. Namespaces and attributes can be defined having labels for the ease of use of the metadata editor. Additionally,

attributes can be defined with ”display filters” which transform the attribute value stored in the iCAT database to a more human-readable form. Currently only regular expression display filters are implemented but others are planned such as a GenQuery filter for translating an attribute with a GenQuery and a JSON filter for transforming JSON encoded attributes into a more visual form for display purposes. A validator interface is planned for the metadata editor enabling client-side validation of metadata entries to the iCAT database e.g with the defintion of allowed values for the attribute in the metadata schema the editor would present the user with a drop-down list of acceptable values. Figure 3 views the metadata manager as currently implemented.

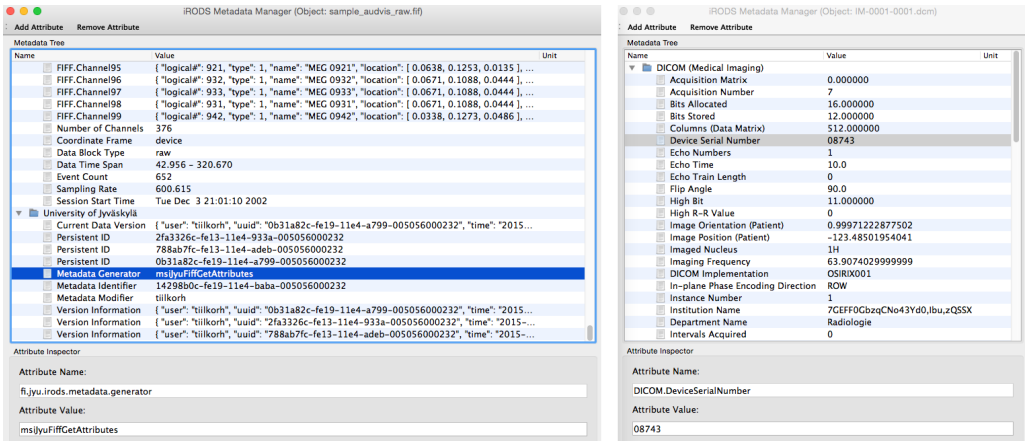

Figure 3. iRODS Metadata Manager windows in the client application.

At the core of the client application is an object-oriented interface for iRODS which wraps around the native C/C++ iRODS API. It is planned that these iRODS specific interface classes are to form an object-oriented C++ iRODS framework. Below is an example of building and executing a GenQuery for retrieving metadata AVU triplets for either a data object or a collection. The code sample is taken from the metadata model class of the client application.

```
Kanki::RodsGenQuery metaQuery(this->conn);
int status = 0;
if (this->objDatum->objType == DATA_OBJ_T) {
    metaQuery.addQueryAttribute(COL_META_DATA_ATTR_NAME);
    metaQuery.addQueryAttribute(COL_META_DATA_ATTR_VALUE);
    metaQuery.addQueryAttribute(COL_META_DATA_ATTR_UNITS);
}
else if (this->objDatum->objType == COLL_OBJ_T) {
    metaQuery.addQueryAttribute(COL_META_COLL_ATTR_NAME);
    metaQuery.addQueryAttribute(COL_META_COLL_ATTR_VALUE);
    metaQuery.addQueryAttribute(COL_META_COLL_ATTR_UNITS);
}
// add a query condition for object name
metaQuery.addQueryCondition(this->objDatum->objType == DATA_OBJ_T ? COL_DATA_NAME : COL_COLL_NAME,
                            Kanki::RodsGenQuery::isEqual, this->objDatum->objName);
// if we are querying a data object also specify collection path
if (this->objDatum->objType == DATA_OBJ_T)
    metaQuery.addQueryCondition(COL_COLL_NAME, Kanki::RodsGenQuery::isEqual, this->objDatum->collPath);
// execute genquery and get status code from iRODS API
if ((status = metaQuery.execute()) < 0) {
    // error reporting code
}
else {
    std::vector<std::string> names = metaQuery.getResultSet(0);
    std::vector<std::string> values = metaQuery.getResultSet(1);
    std::vector<std::string> units = metaQuery.getResultSet(2);
}
```

DISCUSSION

Our main concern with iRODS is related to metadata management – both in terms of metadata quality assurance and the scope of the supported metadata structures. It has been observed that in general, at least 5% of the information present in manually created databases is erroneous [6]. Lessons learned in the institutional repositories domain from self-archiving of publications should be taken into account [7] - researchers should not be responsible for filling metadata fields alone, but as a collaborative process assisted by librarians. On one hand, it is important to allow researchers to edit the metadata entered into the system to get the first-hand insight to the datasets, but it is up to the librarians to ensure that the metadata is in a consistent form, and, if necessary, to "clean" the metadata afterwards. An essential requirement is that the metadata resides in a centralized repository such that two-way synchronizations (and in particular duplicate manual entries) are kept at minimum (i.e. master data is managed) and metadata is reused when possible. Features of our iRODS infrastructure facilitate this goal by preserving information about the latest metadata update (user, timestamp and the metadata UUID), extensible validation functionality, and cascading collection-level metadata to data objects. Practices that yet need to be implemented include duplicate detection, metadata batch editing, and delete/replace on list-like metadata – an effective way to clean up records with misspellings, but to be used with caution [8].

Scoping the supported metadata structures is related to metadata quality. If one is confined to a standard minimum metadata set, object-specific metadata can be perceived as a small set of plain-text fields, resulting to little attention in metadata validation. This alone can be a quality problem if conventions are not followed – especially if data is aggregated from multiple sources [7]. However, depending on the domain, essential metadata may be much more involved, containing diverse compound fields (e.g. MARC in the library domain), or multiple entity classes that may refer to each other. An example is the domain of cultural heritage, where a trend of shifting from item-centric cataloging (physical objects as the primary entities) to event-centric documentation (e.g. CIDOC CRM – events related to the objects) is taking place [9]. Individual fields may be too coarse-grained to represent events, but compound fields specified in a JSON-like structure may be part of the solution. For selected fields, utilization of ontologies (e.g. controlled keywords in the Finnish national Finto service) and other external sources (e.g. name disambiguation with ORCID identifiers) becomes relevant, but needs additional development. One prospect might be declaring special data objects *contained in* iRODS as internally controlled authorities for recurrent, shared metadata (represented as a look-up list), akin to the solution applied in DSpace-CRIS[9]: entities function as authorities for item metadata [10].

As a development framework, Qt provides several benefits compared to both OS-specific (i.e. iExplore [11]) or even web-based solutions. Unified look and feel could be accomplished with other languages or frameworks, but C++ -based compilation and direct linking to iRODS and system libraries provides the best possible performance. Web clients such as iDrop[10] contain useful functionality from data access perspective (no need to install additional software) and we except them to be used for some use cases. However, we argue that a purely web-based client is insufficient for more involved data management (i.e. performing computation on multiple versions of the dataset with close integration to local filesystem). iDrop does not provide schema-specific validation which is a problem from data quality perspective. Another problem with web-based interfaces is – despite recent HTML5 improvements – that file management cannot be implemented with explorer -like capabilities. Even though it is possible to implement drag'n'drop support from local filesystem to browser, it would be limited to file transfer. Mass edit functions, versioning, or 2-way synchronization between web-based view and local filesystem would need an additional plug-in or client application. Therefore, a cross-platform native client application is a critical factor to improve the utilization rate of the system since most users *already expect* a user experience as streamlined (but lacking in metadata, validation, or security aspects) as in popular cloud-based file-sharing services such as Dropbox or Google Drive.

[9] http://cineca.github.io/dspace-cris/

[10] https://github.com/DICE-UNC/idrop

CONCLUSION

Future prospects with our IT infrastructure development project include increased integration – both on the storage and application levels. JYU iRODS could be used as a general purpose storage middleware beyond research data for e.g. the JYU digitization center with metadata imported from the book scanner workflow. The institutional repository JYX would also benefit from iRODS storage in contrast to typical filesystem-based asset store. Potential application-level integrations include connections from iRODS to Dataverse for publishing datasets, and the Current Research Information System (CRIS – currently in procurement at JYU [12]) . Whereas an institutional repository, iRODS or Dataverse are used to store the *outputs* of a research project, a CRIS system provides information about the research projects themselves. This includes metadata related to publications and project-related documentation (e.g. research plans, funding decisions). A CRIS could be used as a data hub combining information about research infrastructures, projects, and outputs – used for aggregating metadata from other sources (i.e. bibliographic databases) as well as feeding it to other systems, such as a data warehouse, an institutional repository, or an iRODS grid.

The development of our client is still at early stages. The iRODS infrastructure and our client have been presented to other Finnish HEIs and the National IT Center for Science CSC at the National IT Days for Higher Education (*IT-päivät* in Finnish) and other occasions. Our solution has provoked interest, showing potential to be of use in other universities as well as IT services operated by CSC. We intend the development process to be a collaborative effort and plan to publish the code under an open source licence. We welcome suggestions regarding the features for the UI, validation, and data description format and hope that the software will be utilized in other institutions.

ACKNOWLEDGMENTS

The authors thank the Head of IT Management at the University of Jyväskylä, D.Sc. Antti Auer, for his contribution on research data management policies at JYU, enabling management support for the development project.

REFERENCES

[1] A. Neuvonen *et al.*, "ATT-vaikuttavuusselvitysryhmän raportti [ATT effectiveness working group report]," Tech. Rep., 2015. [Online]. Available: https://confluence.csc.fi/pages/viewpage.action?pageId=45389145

[2] P. Olsbo, "Institutional repository as a center of services," 2012, poster at IFLA World Library and Information Congress. [Online]. Available: http://www.libraries.fi/en-GB/finnishlibraryworld/presentations/

[3] A. Auer and S.-L. Korppi-Tommola, "Principles for research data management at the University of Jyväskylä," Tech. Rep., 2014. [Online]. Available: https://www.jyu.fi/tutkimus/tutkimusaineistot/rdmenpdf

[4] T. Walters, "Assimilating digital repositories into the active research process," in *Research Data Management – Practical Strategies for Information Professionals*, J. M. Ray, Ed. Purdue University Press, 2014.

[5] Y. Kawai and A. Hasan, "High availability iRODS system (HAIRS)," in *Proceedings of iRODS User Group Meeting 2010*. Data Intensive Cyberinfrastructure Foundation, 2010.

[6] A. van den Bosch, M. van Erp, and C. Sporleder, "Making a clean sweep of cultural heritage," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 54–63, 2009.

[7] J. W. Chapman, D. Reynolds, and S. A. Shreeves, "Repository metadata: Approaches and challenges," *Cataloging & Classification Quarterly*, vol. 47, no. 3-4, pp. 309–325, 2009.

[8] M. Nurminen and A. Heimbürger, "Representation and retrieval of uncertain temporal information in museum databases." in *Information Modelling and Knowledge Bases XXIII*. IOS Press, 2012.

[9] A. Häyrinen, "Open sourcing digital heritage – digital surrogates, museums and knowledge management in the age of open networks," Ph.D. dissertation, University of Jyväskylä, 2012.

[10] S. Mornati and A. Bollini, "DSpace-CRIS: an open source solution," 2013, presented at euroCRIS Membership Meeting. [Online]. Available: http://www.eurocris.up.pt/?page_id=13

[11] B. Zhu, "iExplore for iRODS distributed data management," in *Proceedings of iRODS User Group Meeting 2010*. Data Intensive Cyberinfrastructure Foundation, 2010.

[12] M. Nurminen, "Preparing for CRIS: Challenges and opportunities for systems integration at Finnish universities," 2014, poster at Open Repositories 2014.