Archie: functional design

Archiving for Archaeologists made EASY

Wouter Kool, Data Manager NEXUS1492

Background

The NEXUS1492 project (and a couple of related projects - in total some 40 researchers - investigates the impact of colonial encounters in the Caribbean. It does so through new archaeological and historical research, as well as current interdisciplinary techniques (for instance network analysis, isotope analysis and DNA analysis).

Since 2015, the Faculty of Archaeology has made it compulsory for running research projects to deposit their data to the KNAW/DANS archive EASY, complying to the (Guidelines for Archaeological Data) as much as possible. This means a researchers have to catch-up to meet these requirements.

In order to make this as EASY as possible, the plan has arisen to develop a tool to support these activities.

Also, this tool can aid the researchers to supply their data to the Internal Data Repository to be developed for the Caribbean Archaeology department as a whole (and, if there appears to be a wider demand, for the entire Faculty of Archaeology). This system is to be integrated at a later stage with the data archiving infrastructure currently under development by the ISSC.

General purpose

The purpose of the tool is to support researchers with depositing their data to the above repositories. It helps them to create the necessary metadata in a user-friendly way, pre-filling the metadata files where necessary.

Wireframes

This document should be used in conjunction with a set of wireframes. The wireframes should be viewed as indicative of the functionalities. Currently it lists a stand-alone GUI-app, but exact implementation depends on the technical implementation to be decided upon (see Dependencies).

The wireframes are created using Balsamiq Mockups and will be supplied with this document in PDF format. A video recording the workflow is also available on request, but large to distribute.

Dependencies / Risks

NEXUS project, At the point of writing, the data management infrastructure of the University is under active development and there are a number of uncertainties.

iRODS has been chosen as a middleware solution, but the exact implementation choices are not known.

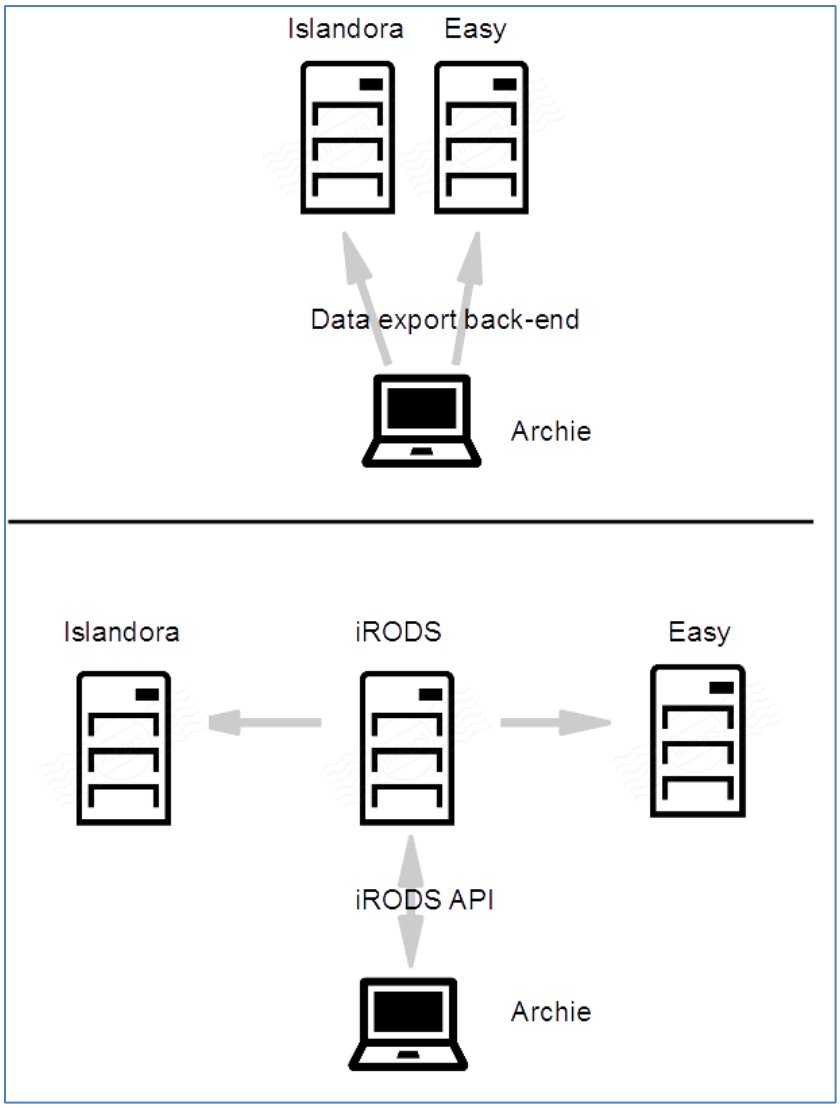
Because the archiving responsibilities, as well as the Data Repository will continue to exist after NEXUS1492 has finished, changes in this environment should be reckoned with in order for the tool to be able to (be adapted to) function in this environment. As a counter-measure the following activities will be undertaken:

- Request any available information about the infrastructure-to-be-developed.
- Create a stand-alone tool, preventing for instance tight coupling to Islandora.[1]
- Keep the back-end flexible, to be able to accommodate multiple output formats.
- Use common programming languages that preferably have a native iRODS APIs, so there is a large choice of interfaces with iRODS.

System Context

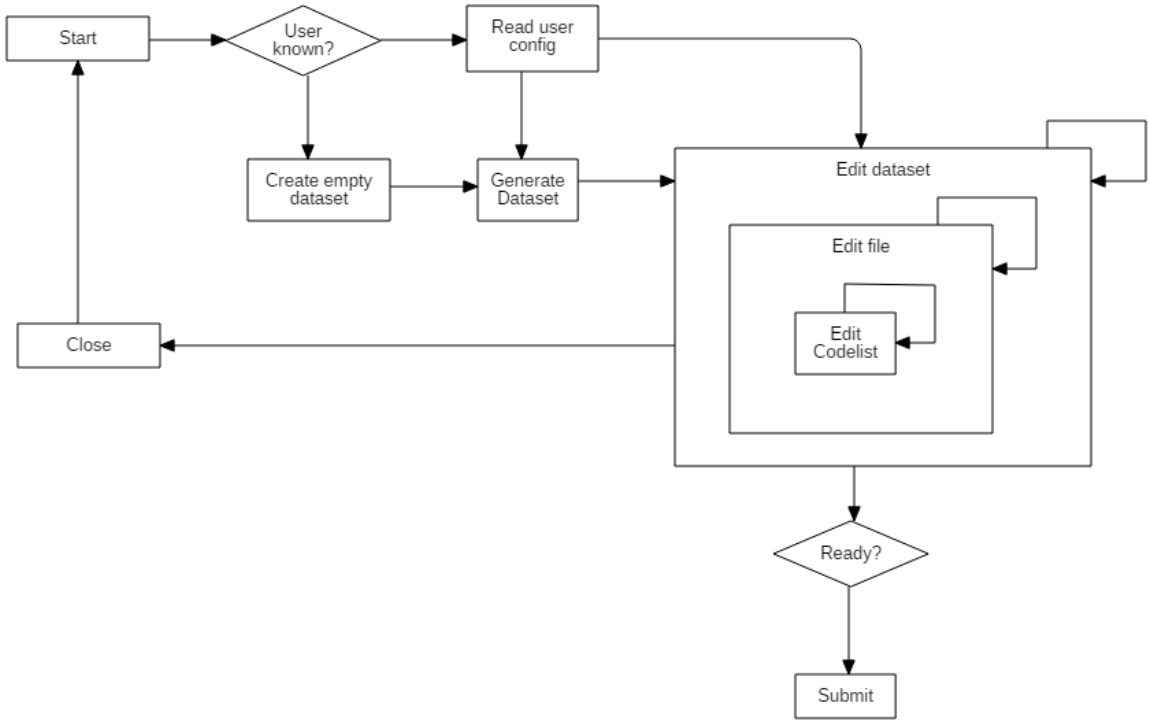
Depending on the decisions in the environment mentioned above, the system context will change overtime in an unknown way. To get a discussion started, we have drawn a possible scenario. The first one might be the situation for phase 1, the second one, another might be the final situation.

[1] This prevents building upon the data management module for Islandora developed by the University of Prince Edward island which in itself looks very promising: https://data.upei.ca/islandora/object/data:4



Workflow description

The application implements the following workflow.



Functional Requirements

ID	Name	Description	Priority
FR1	Define dataset	A number of datasets can be defined. Each dataset can be linked to a directory on the file system all directories inside this directory belong to the file set.	Must
FR2	Specify dataset metadata	A number of metadata fields can be filled-in	Must
FR3	Specify file metadata	For every file a number of properties can be filled-in (this list of properties corresponds to the KNAW-DANS Guidelines for Archaeological Metadata).	Must
FR4	Specify Codelist	For every file of a given range of types (MSExcel, MSAccess, SQL Dump, SPSS, etc.), a code list can be filled-in, containing all tables/spreadsheets, used columns /coded values and scales), for databases the relationships between tables can be defined	Must
FR5	Prefill file metadata	File metadata will be prefilled by the application as much as possible using the file system and extraction from the embedded metadata in the file. This data is shown to the user and can be edited afterwards	Must
FR6	Prefill code lists	From a small number of standard file types than the list in FR3 (only MSExcel and MSAccess, SQL Dump) and some structured way of embedding codelists in the file, a code list can be prefilled. This data is shown to the user and can be edited afterwards	Must
FR7	Configure mother project	It is possible add or edit a mother project. A number of dataset metadata elements can be	Nice

		filled in. The mother project is added to the configuration	
FR8	Specify submitter	The submitter has to be identified. (this can probably be done automatically by identifying the current user on the system)	Must
FR9	Save submission data	The submission data is stored in a persistent way on the file system	Must
FR10	Resume submission activities	The submission data is retrieved from the file system and presented in the user interface.	Must
FR11	Choose mother project	The user can select a mother project for his submission. This will cause some dataset metadata to be prefilled when he creates a new dataset.	Nice (now only NEXUS1492. Might be hard-coded
FR12	Specify repository	Select a repository from the predefined list of repositories	Nice
FR13	Add/edit repository	Add a new repository to the list of repositories. It is possible to define a submission protocol, an upload url, packaging format and a metadata format	Nice
FR14	Generate submission package	Generates a submission package for a given repository according to the given packaging and metadata format. At least for one repository: Islandora. And probably for KNAW Dans - sword format.	Must
FR15	Submit	Submit the data to the repository. A username and password has to be filled-in.	Nice (Might be done by batch delivery)
FR16	Read configuration	On startup, the configuration is read containing predefined mother projects and repositories	Nice

Non-Functional requirements

ID	Name	Description	Priority
NFR1	Graphical User interface	See wireframes, for a rough idea. Exact implementation will depend on technical dependencies.	Must
NFR2	Cross platform	Should at least work on Windows/Mac (recent versions), or a web interface to be accessed from any system.	Must
NFR3	Response times	There should be reasonable response times, for instance when reading generating the datasets or reading the data on onstartup	Must
NFR4	Flexible backend	The backend should be flexible to accommodate future choices in the environment (see Dependencies). First, it should be reckoned with that two output formats should be developed (1) KNAW/DANS Guidelines for Archaeological metadata and (2) Islandora Ingest requirments. Preferably there should be a single persistent format that is able to produce these two formats in a flexible and maintainable way (see Technical	Must

		Requirements).	
NF5	Installation	If installation by the end user (if any) should be a	MUST

Technical requirements

ID	Name	Description	Priority
TR1	Platform	The application might be web-based or stand-alone (provided it functions cross-platform and it is not necessary to	
TR2	Programming language	To be decided, up front preference for JAVA or Python.	
TR3	Metadata extraction	Apache Tika[2] seems like a suited component for for metadata extraction	
TR4	Persistance / configuration	Use XML (preferably without namespaces)	
TR5	Conversion to repository format	Use XSLT for XML conversion	
TR6	Repository backend	For first iteration, DANS output[3] is demanded at least, as well as Islandora[4] is to be developed. If the NEXUS1492/Islandora project will develop a backend to EASY within a timeframe that corresponds to the NEXUS1492 timeframe only a an Islandora backend is needed. For future versions, integration with one of the iRODS APIs might be needed.[5]	

Data model

The application implements the following logical data model. It is not a database design (making no choices about private keys and foreign keys etc.).

[2] http://www.tutorialspoint.com/tika/tika_metadata_extraction.htm

[3] Probably using the packaging format (Dublin Core / BagIt) specified by the SWORD interface: https://easy.dans.knaw.nl/schemas/docs/sword-v1-packaging.html

[4] Description of the Islandora batch ingest format: https://wiki.duraspace.org/display/ISLANDORA113/How+to+Batch+Ingest+Files

[5] For an overview of iRODS APIs, see: http://irods.org/wp-content/uploads/2016/06/technical-overview-2016-web.pdf and https://www.surf.nl/binaries/content/assets/surf/nl/2015/20150428-presentatie-an-overview-of-irods-clients---ton-smeele.pdf (this is an older document so the situation might be changed). Separate APIs have neat documentation on Github.

Data Model1::ERDDiagram1