RODA-in

A generic tool for the mass creation of Submission Information Packages

José Carlos Ramalho
Dep. Informatics
University of Minho
Portugal
jcr@di.uminho.pt

André Pereira
KEEP SOLUTIONS Lda
Portugal
apereira@keep.pt

Miguel Ferreira
KEEP SOLUTIONS Lda
Portugal
mferreira@keep.pt

Luís Faria
KEEP SOLUTIONS Lda
Portugal
lfaria@keep.pt

Abstract—**RODA-in is an offline tool designed to easily create thousands of SIPs with gigabytes of data in an easy to use way. This is possible by using aggregation rules, which map files and folders to SIPs, and metadata association rules, which add metadata to the created SIPs. The basic workflow can be defined in a sequence of easy steps where the user starts by selecting the folders to be archived and then chooses which patterns will be used to transform the data in SIPs. As an optional step, it's possible to edit the generated SIPs to either enrich them or fix exceptions to the rule. Lastly, it's possible to export to two different formats: BagIt and E-ARK SIP. In this paper we present and discuss all the decisions and ideas taken to implement RODA-in like which workflow should be used, what aggregation and metadata association options are currently implemented, how the metadata templating system works and which other features can be used to enrich the SIPs.**

Keywords-Digital Preservation; Open Archival Information System (OAIS); Submission Information Package (SIP); Archives; Electronic Records Management System (ERMS); Interoperability.

I. INTRODUCTION

The advent of the digital revolution has introduced a profound change in the way archives operate. Records are now born-digital, and the well-established processes of transferring content from producers to archives do not cope well with the specifics of digital records.

Although concepts in this area are already matured, e.g. the concepts of submission information packages (SIP) and ingest processes are defined on the Reference Model for an Open Archival Information System [1], there has been a lacking of open source tools to support the Producers in the creation of such packages. One common approach is the development of specific integrations between Electronic Records Management Systems (ERMS) in use at the Producers side and the archive, allowing to create SIPs directly from these systems and submit them to the archive's ingest workflow. However, these integrations focus only on the usual suspects, failing when more heterogeneous systems are in place or when Producers use niche systems to support their records management activities. Furthermore, a considerable number of Producers do not use a document management system at all and simply manage their records on a shared folder on the local network.

When systems integration is not possible due to lack of support, expertise or budget, professionals create SIPs manually. While many archival systems provide tools to help with the creation of SIPs, e.g. Web forms or downloadable software, tools that are designed to create SIPs one-by-one are highly inefficient and even unusable in contexts where a Producer has a high volume of information that needs to be sent to the archive.

This work describes a novel approach to deal with the scenario where a Producer has a high volume of information managed directly on the file system. The same approach may also be applied to Producers that have ERMS that enable users to export content and metadata to the filesystem.

The proposed approach is based on an offline desktop tool called RODA-in 2.0 that allows Producers to massively select files and folders and aggregate them as SIPs, while at the same time defining rules for associating descriptive metadata and documentation. Aggregation rules are based on common information management patterns found in Producer organisations. These rules will continue to be expanded and improved as real world use of the tool provides more useful feedback.

Albeit its name, and its clear association with RODA project[1], RODA-in 2.0 was created to be independent of any implementation of a digital repository. It is currently capable of creating SIPs in the E-ARK SIP format [7] and the Library of Congress BagIt package format[2].

The tool is currently being tested at the pilot sites of the E-ARK project which includes several prominent European National Archives and by Producers that are part of the Digital Continuity network of the Portuguese National Archives.

[1]http://www.roda-community.org/

[2]http://www.digitalpreservation.gov/documents/bagitspec.pdf

This paper's remainder comprises three sections: section 2 describes all relevant concepts to the subject, section 4 describes RODA-in architecture, possible use case scenarios and all the decisions taken in its development, section 5 closes the paper with conclusions and currently undergoing future work.

II. Concepts

The work presented in this article is relevant to many domains, not just Archives or Libraries. Environments where information is managed by one organisation is expected to be prepared, packaged and submitted to another organisation for long-term preservation is a prime candidate to be a user of this tool.

Archival Information Collection (AIC): An Archival Information Package whose Content Information is an aggregation of other Archival Information Packages [1]. Note that these collections can be used for Classification of records.

Archival Information Package (AIP): An Information Package, consisting of the Content Information and the associated Preservation Description Information (PDI), which is preserved within an OAIS.

Classification [scheme]: In records management, the systematic identification and arrangement of business activities and/or records into categories according to logically structured conventions, methods, and procedural rules represented in a classification system [4].

Content Data Object: The Data Object, that together with associated Representation Information, comprises the Content Information. Data Object is either a Physical Object or a Digital Object. Digital Object is an object composed of a set of bit sequences [1].

Content Information: A set of information that is the original target of preservation or that includes part or all of that information. It is an Information Object composed of its Content Data Object and its Representation Information [1].

Description level (a.k.a. Level of description): The position of the unit of description in the [classification] hierarchy [5].

Descriptive Information: The set of information, consisting primarily of Package Descriptions, which is provided to Data Management to support the finding, ordering, and retrieving of OAIS information holdings by Consumers [1].

File: a named and ordered sequence of bytes that is known by an operating system. A file can be zero or more bytes and has a file format, access permissions, and file system characteristics such as size and last modification date [3].

Intellectual entity: A set of content that is considered a single intellectual unit for purposes of management and description: for example, a particular book, map, photograph, or database. An intellectual entity can include other intellectual entities; for example, a Web site can include a Web page; a Web page can include an image. An intellectual entity may have one or more digital representations [3].

Producer: The role played by those persons or client systems that provide the information to be preserved [1].

Representation: The set of files, including structural metadata, needed for a complete and reasonable rendition of an Intellectual Entity. For example, a journal article may be complete in one PDF file; this single file constitutes the representation. Another journal article may consist of one HTML[3]file and two image files; these three files constitute the representation. A third article may be represented by one TIFF[4]image for each of the 12 pages plus an XML[5] file of structural metadata showing the order of the pages; these 13 files constitute the representation [3].

Representation Information: The information that maps a Data Object into more meaningful concepts. An example of Representation Information for a bit sequence which is a FITS file might consist of the FITS standard which defines the format plus a dictionary which defines the meaning in the file of keywords which are not part of the standard. Another example is JPEG software which is used to render a JPEG file; rendering the JPEG file as bits is not very meaningful to humans but the software, which embodies an understanding of the JPEG standard, maps the bits into pixels which can then be rendered as an image for human viewing [1].

Submission Information Package (SIP): An Information Package that is delivered by the Producer to the OAIS for use in the construction or update of one or more AIPs and/or the associated Descriptive Information [1].

III. Related Work

This need for a tool to assist the creation of information packages is not new and during the last years some initiatives emerged.

The U.S. Library of Congress has developed the Bagger tool. The Bagger application assists the user to produce a package of data files according to the BagIt specification. It works like a graphical user interface to the BagIt specification.

BagIt is a hierarchical file packaging format designed to support disk-based storage and network transfer of arbitrary digital content. A "bag" consists of a "payload" (the arbitrary content) and "tags", which are metadata files intended to document the storage and transfer of the bag. A required tag file contains a manifest listing every file in the payload together with its corresponding checksum. The name, BagIt, is inspired by the "enclose and deposit" method, sometimes referred to as "bag it and tag it" [2].

University of Kentucky also developed a tool based on the BagIt specification: Exactly.

Exactly is a simple and easy to use application for remotely and safely transferring any born-digital material from a sender to a recipient.Built on work originally begun by the Gates Archive, AVPreserve and the Louie B. Nunn Center for Oral History, University of Kentucky Libraries developed Exactly to meet the growing need for archives to acquire born digital content directly from donors and to begin the activities of

[3] HTML means Hypertext Markup Language.
[4] TIFF means Tag Image File Format.
[5] XML means Extensible Markup Language.

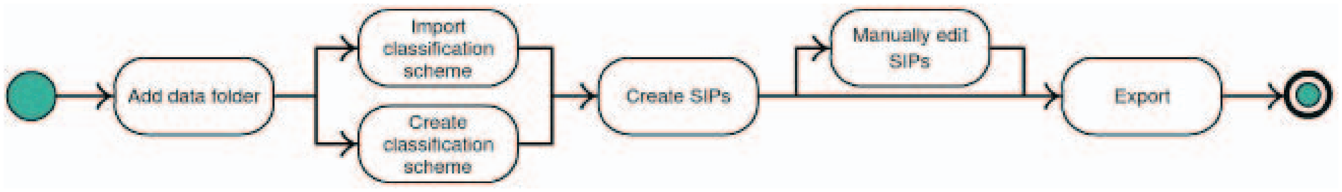

Figure 1: The basic workflow of RODA-in

establishing provenance and fixity early in the process of acquisition.

Estonian National Archives developed their own specific tool: Universal Archiving Module (UAM).The universal archiving module is a software created by the Estonian National Archives for the preparation and transfer of digital documents extracted from electronic records managements systems. Use of UAM requires the ability of an institution's ERMS to export documents and their metadata in XML format.

These are the main similar tools that are being supported by some institution and used in some real case scenarios. We can notice that BagIt has a strong presence and is being used as the support format by many tools. If we execute a quick search is easy to find many small applications and code libraries able to deal with the BagIt specification. The problem is that most tools are focused on metadata creation or package transfer. Our focus is massive creation of packages, we want to automate the creation of thousands of packages at a time. This brings the problem to a new dimension that must be taken care of differently.

In other hand, we were working in the E-ark project that intends to establish european guidelines for this context. In this line of work E-ark created an EARK SIP specification. Our tool supports this new specification and BagIt since it is still used in many institutions.

IV. RODA-IN APPROACH

The approach taken was to create an offline application for producers and archivists to create SIPs ready to be submitted to an Open Archival Information System (OAIS) [1].

The file system works as the source of information which is used in semi-automated SIP creation based on rules and heuristics.

The workflow can be described in five easy steps:

1. Choose the folder that holds the data to be archived;
2. Import or create a new classification scheme;
3. Create a rule to transform the input data to SIPs;
    a. Choose SIP-creation aggregation rule from the 4 existing options;
    b. Choose the descriptive metadata association rule from the 4 existing options;
4. Manually edit the created SIPs (Optional);
5. Export the SIPs in the desired format;

There are additional steps that the user can take to tweak the output like preventing some files to be added to the SIPs (e.g. Thumbs.db, DS_Store, invisible files, etc.) and editing or adding new descriptive metadata. One of the goals of this tool is to provide a simple approach to enable users to create thousands of SIPs with just a few mouse clicks, but also to enable advanced features that satisfy the needs of more demanding users.

A. *Source file explorer*

The first step of the workflow is to choose which computer files and folders to archive and add them to the application's file explorer. This action will display the folder and its content in a folder tree structure. The folders and files are clearly distinguishable from each other and folders can be expanded in order to inspect its children elements.

Special attention has been dedicated to cope with large amounts of files and folders. User graphical interfaces struggle to display folder contents with millions of items. To overcome this problem, only the first one hundred items are loaded at once and a button has been included to load the next hundred items. Even if not all the items are visible at a given time, these will processed adequately if their hierarchical ascendants are selected for processing.

An item can be in one of three states: normal, ignored or mapped. **Normal** is the default state of the items and the only state from which SIPs can be created.

Items in the **ignored** state will not be used to create packages. They can be ignored based on rules defined in the configuration of the application, or manually using the "Ignore" button. These items are hidden by default, but can be made visible by turning on a menu option.

Mapped items are the ones that have already been added to an SIP. The goal is to implement a strategy we like to call "map and forget" in which users drag n' drop items from their file system to create SIPs after which they disappear from the user interface. This enables users to fully focuse on the remaining items and thus be more productive in their SIP creation project.

Due to the states that an item can be in and the fact that we don't necessarily know the full domain being used (since we don't load the full tree right away to improve performance), an algorithm that assures the file explorer is coherent with the rest of the application was developed. This algorithm uses a collection of paths and its states. Since each path can only have one state at a time, a one-to-one relationship can be used. When an item is added to the file explorer tree, it is also added to this collection. For performance reasons, as stated earlier, the full tree is not visited right after a folder is added to the file explorer, it's only when the SIPs are being created that information of the full tree is accessed.

When getting the state of a path, the collection is checked for associated states. If there are any, the state of the path is the one saved in the collection, but when the path does not exist yet, some more computations are needed to find it. The algorithm starts going up the tree until it finds an ancestor's path that is in the collection. When found, the state of the path is the state of the ancestor.

There are a few rules that were defined to keep the state coherent:

- A folder is ignored only if all of its children are ignored as well;
- A folder is mapped if at least one of its children is mapped and no child has the normal state;
- A folder is normal if at least one of its children has the normal state;
- A state can only be set as mapped or ignored if the previous state was normal. We cannot have ignored to mapped changes or vice-versa.

With these rules in mind, additional steps are needed when adding or updating a path's state, since this can mean that one of its ancestors may change its state. When mapping or ignoring a file a shortcut can be taken by only applying the state to the paths already in the collection (since all the other will infer their state by using the information of their ancestors), but the same cannot be performed on the other way around, when a path is set as normal when it was mapped or ignored. This is because the user performed operations are unknown, therefore it can't be safely deduced that all the children of a path are to be set as normal as well. When removing the mapped state of a set of paths, it must be assured that only those paths are affected (and correctly update the ancestor's state), since there could be other mapped paths that would be affected if the same kind of shortcut was used. As a consequence of this, removing SIPs is a task that takes much longer to perform than the creation process. But as removing previously created SIPs are a much less common task than creating the SIPs, this disadvantage does not have a big impact.

B. *Definition of the classification scheme*

This optional step allows the user to define how the SIPs will be organised in a classification hierarchy. This classification can be imposed by the archive, created voluntarily by the Producer, or a mix of the two. In the later case, the hierarchy is imposed by the Archive up to a given point and then further defined by the Producer. There is also the option of not using a classification scheme. In this scenario, SIPs are created without holding any information about where they fit in the classification hierarchy. It is up to the Archive to decide how the SIPs should be classified after the ingest process.

To enable classification schemes, a number of features were added to the tool, namely the ability to import a full classification hierarchy (only RODA specific format is supported at the moment), and the ability to add, edit and remove nodes of the classification scheme, as well as moving SIPs from and to this classification hierarchy[6].

C. *Creation of SIPs*

The file explorer and the classification scheme are fundamental to the SIP creation workflow implemented by RODA-in. The first is used to select the data to be included in the SIPs, and the second is used to define the collections or series where the data should be placed once it's accepted by the repository.

To create SIPs, the user has to choose an aggregation rule. Start by selecting the files and folders that hold the content to be included in the SIPs and drag n' drop them on a node of the classification hierarchy. This node can either be a classification scheme node, a SIP or the root level (i.e. no specific node). After this operation, the user is presented with a set of options that will shape the way the SIPs are created.

D. *Aggregation rules*

There are four content aggregation options that should satisfy the needs of most Producers. More options will be added as the user community unveils other prominent patterns.

Option 1: One SIP for each of the selected files or folders

This option creates one SIP for each selected file or folder. It is mostly useful when Producers organise records on a per folder-basis, i.e. each folder contains all the files that are relevant to a given record.

For example, if the input is composed of 3 files and 2 folders, this association option will create 5 SIPs, one for each of the 3 files and one for each of the 2 folders. This is a N to N mapping between the input data and the output SIPs.

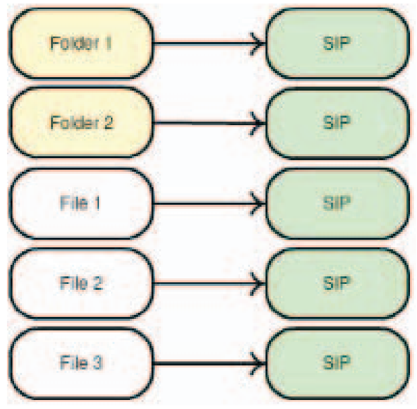

Figure 2: Folder and files mapping to SIPs using Option 1

[6]The drag n' drop system allows the users to select one or more items from the tree and move them to another node or to the outside of the tree.

Option 2: One SIP containing all selected files and/or folders

The second option creates one and only one SIP containing all the files and folders from the input. It's useful when the Producer wants to create one single SIP containing a large number of files and folders (potentially the entire year worth of records or a hard drive).

For example, if the input is composed of 5 files and 2 folders, this association option will create 1 SIP containing the entire set of selected items.

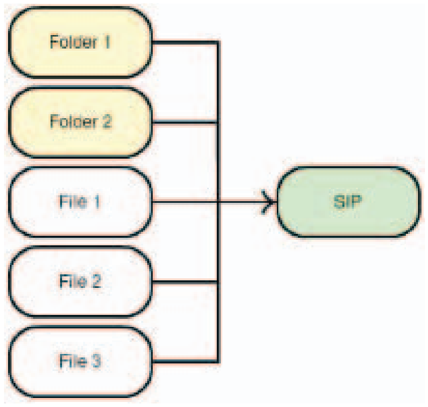

Figure 3: Folder and files mapping to one SIP using Option 2

Option 3: One SIP for each file under the selected folder(s)

This option creates one SIP for each file included in any of the selected items. The files are found recursively inside the folders, creating a set of SIPs equal to the number of files found. It's useful when each individual file represents an entire record.

For example, if the input is composed of 3 files and 2 folders with 10 files each, this association option will create 23 SIPs, one per each selected file and per each of the 10 files under each of the 2 folders.

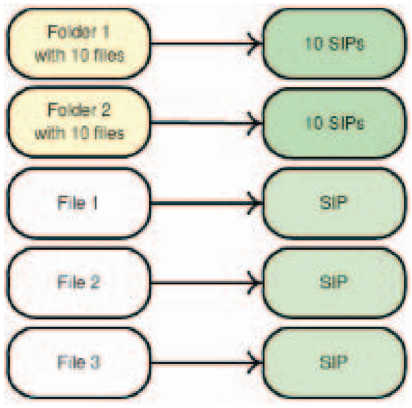

Figure 4: Folder and files mapping to SIPs using Option 3

Option 4: Create classification scheme and SIPs based on folder structure

This option creates a classification scheme and SIPs containing data based on the original folder structure that holds the data.

This option is useful when the Producer maintains a well organized folder structure that resembles the desired output classification scheme. Leaf folders that contain files are treated as SIPs while ascending folders without files are used to create nodes in the classification scheme.

The rules in place are as follows:

- If a folder has sub-folders and no files, it creates a node on the classification scheme;
- If a folder only contains files and no folders, the whole folder will become a SIP;
- If a folder has mixed content, i.e. files and sub-folders, each file will be a SIP and each sub-folder will be a node in the classification scheme.

1) Patterns for metadata association

SIPs may contain data and metadata. The metadata is included in each SIP by means of association rules. These association rules define how descriptive metadata selected from the file system and included inside each SIPs.

There are currently four available metadata association options, but the implementation enables more to be added in the future.

Option 1: Metadata based on a template

In some cases, Producers may not have metadata and they expect the SIP creation tool to facilitate this activity. RODA-in comes with a metadata templating system that enables users to add their own descriptive metadata schemas to be rendered as easy-to-use input forms.

The templating system implemented by RODA-in enables the user to define the base structure of the descriptive metadata to be included in SIPs, define fields to be displayed and edited in the user interface and fields that should be filled with a predefined value (e.g. institution name) or filled by a auto-generated value (e.g. current date).

By default, RODA-in comes with templates for EAD 2002, EAD 3 and Dublin Core that users are free to change and adapt to their own needs.

Some fields from the template are pre-filled by the application:

title: The title is inferred using the description item's content, like the name of a folder or file in a SIP;

now: Sets the value as the current date in the format yyyy-MM-dd;

id: Generates a random UUID;

level: Sets the value as the item's description level;

parentid: The UUID of the item's parent.

Option 2: Same metadata in every SIP

This option should be used when the user wants to apply the same metadata file to every package created. The input must be the metadata file that will be added to each of the SIPs to be created.

Option 3: Load metadata from data folder

This option takes a user-defined pattern[7] to match file names under the data folder and add the identified files to the SIP as descriptive metadata. This is the only option where more than one metadata file can be added to the same SIP. When no files match the pattern, the SIP will be created without metadata.

Option 4: Load metadata from a folder

When all the metadata files are located in a single folder, the Producer can use this option to selected and include one metadata file in each of created SIPs. The tool will try to match the name of metadata file with the name of the SIP (based on the data folder name). For a match to occur, the name of the metadata file (without the extension) and the SIP must be the same. Just like the metadata option number 3, if no matches are found, no descriptive information will be added to the SIP.

E. Handling exceptions to the rule and SIP enrichment activities

After creating SIPs, users can opt to manually enrich the SIPs or handle exceptional cases manually. This is an optional step, nevertheless it contains some important features that should not be overlooked.

When SIPs are created using the metadata template option, users are able to edit metadata via an input form. Additionally, the XML source of the metadata file can also be displayed and edited on the user interface.

In addition to editing metadata files, it's possible to add and remove metadata files. There are three options available when adding a new metadata file: 1) new metadata based on an existing template, 2) new metadata based on an existing file, and 3) an empty metadata file. To avoid conflicts, two metadata files of the same description item cannot have the same name, so new files are rejected if this happens. As a consequence of this, two files based on the same template cannot be present in the same description item.

Besides changing the descriptive metadata in the SIP, the Producer can also edit the data content of the SIP. The content is divided in "representations" and the user is able to add and remove these as long as there is at least one representation left in each SIP.

Removing files and folders from the SIP content is also possible. These files will return to the "normal" state and will become available in the file explorer to be included in another SIP. A drag n' drop mechanism of the data folder structure enables rearranging the content of the SIP and move files between representations.

The RODA-in tool also enables adding files to the "documentation" folder as defined in the E-ARK SIP format. This acts as a special form of Representation Information (as defined by OAIS) [1].

F. Exporting SIPs

The last step in the SIP creation process is to export the mapped SIPs to an output format to be submitted to the repository for ingest. It's important to note that RODA-in does not check if the SIPs are valid before exporting them and this process should be performed on the repository or with an external tool. Currently there's two export formats supported: BagIt and E-ARK SIP.

1) The BagIt format

BagIt is "a hierarchical file packaging format for storage and transfer of arbitrary digital content. A 'bag' has just enough structure to enclose descriptive 'tags' and a 'payload' but does not require knowledge of the payload's internal semantics."[8].

The minimum structure of a valid bag is described in Figure 6. The base directory may have any name.

```
<base directory>/
  -- data/
    -- [payload files]
  -- [optional tag directories]/
    -- [optional tag files]
  -- bagit.txt
  -- manifest-<algorithm>.txt
  -- [optional additional tag files]
```

Figure 5 : Basic bag structure

The "bagit.txt" file declares the BagIt version and the encoding of the tag files. The "manifest-<algorithm>.txt" file is a payload manifest that lists all the payload files and their checksum using an algorithm like MD5 [9] or SHA-1 [8]. The data folder is where the payload files are saved.

One of the more important optional tag files is bag-info.txt because that is where all the bag's metadata will be. Since there is no concrete location in a bag to send metadata in, RODA-in dumps all the metadata files from the SIPs internal representation to this file in a key-value format, where the key is "metadata.<file_name>" and the value is the file's content. It is up to the repository to parse this information and divide it in files again.

There is some information supported by RODA-in that is not directly mapped when exporting to the BagIt format since there is no adequate placeholder for "representations" or "documentation". Nevertheless, this format can cope with the simpler cases where this complexity is not needed and would not even be supported on the target digital repository.

2) The E-ARK SIP format

The E-ARK SIP was developed in the E-ARK project because "there is currently no central SIP format which would cover all national and business needs as identified in the E-ARK Report on Available Best Practices" [6].

A simple description of the E-ARK IP structure is available in Figure 6.

```
base directory/
  -- metadata/
    -- descriptive/
    -- preservation/
    -- other/
  -- representations/
    -- <representation 1>/
        -- <representation 2>/
      -- documentation/
      -- schemas/
      -- METS.xml
```

Figure 6: Basic E-ARK SIP structure

The E-ARK SIP format is more complex than BagIt. It has distinct folders for metadata, data (with support for "representations"), documentation and schemas. The METS.xml is a "mandatory metadata file which includes core information needed to identify and describe the structure of the package itself and the rest of its components" [7]. This file is based on the METS standard and is presented as a profile with 5 main sections:

<metsHdr> - METS header (metadata about the creator, contact persons, etc. of the IP);

<dmdSec> - descriptive metadata (references to EAD, EAC-CPF, etc.);

<amdSec> -administrative metadata (how files were created and stored, intellectual property rights, etc.);

<fileSec> - file section, lists all files containing content (may also contain metadata about files);

<structMap> - structural map, describes the hierarchical structure of the digital object and the whole IP (i.e. object + metadata).

For now, all the metadata is added to the E-ARK SIP as descriptive metadata. The documentation is added exactly as displayed in the application, keeping the full folder structure and all the files. The schemas folder is by default populated with the METS schema file which depends on the XLink schema, which is also added. Moreover, the schemas from the metadata templates used are also added to this folder, given that they are set in the template's configurations.

Each representation also has its own folder structure that is similar to the overall SIP structure, as shown in Figure 7.

```
<representation name>/
  -- data/
    -- [files]
  -- metadata/
    -- descriptive/
    -- preservation/
    -- other/
  -- METS.xml
```

Figure 7: Representation structure

The "METS.xml" file is optional and should be used to handle scalability issues since it describes the structure and content of the representation only. The data folder is where the files that will be archived are stored. If all the metadata is stored at the IP level then there's no need to use the Metadata folder at the representation level. Aside from this folder being optional, RODA-in doesn't yet support adding metadata or documentation for specific representations, only at the IP level.

G. *Other features*

Besides the described features, RODA-in has other components to help the Producer create SIPs like descriptive metadata validation, templates, forms and documentation.

The templating system is based on the widely used semantic templates library: Handlebars[9]. A template is a string that contains any number of tags. Tags are indicated by the double curly braces that surround them: {{person}} is a tag, as is {{#person}}[10]. Each of these tags corresponds to a text field on the form, where the user can set the value that will replace the tag when the metadata file is exported. There are some tags that the application recognizes and automatically sets them with auto-generated values.

[9]http://handlebarsjs.com/

[10]https://github.com/janl/mustache.js

The templates are completely customizable since it's possible to not only edit their content but also add new templates or remove those that are unwanted. All of the user customizable data, like the configurations file, the templates and the logs are in the application folder that can be found in the user's home directory in Unix systems or in the "My documents" directory on Windows. To add a template the user needs to follow some easy steps:

- Add the file with the template to the "templates" folder;
- (optional) Add the schema that validates the template to the "schemas" folder;
- Edit the configuration file.

In the configuration file, the archivist needs to insert data to tell the application which file, title, version and schema should be used in the new template. Another required field is the unique identifier, which in the example below will be "ipres2016".

```
metadata.template.ipres2016.file
    = ipres2016.xml
metadata.template.ipres2016.title
    = The iPRES 2016 template
metadata.template.ipres2016.version
    = 2016
metadata.template.ipres2016.schema
    = ipres2016.xsd
```

After this configuration block has been added, all the user needs to do is add the template to the active templates list and restart the application:

```
metadata.templates =
ead,ead3,dc,ipres2016
```

Provided that the schema is added to the templates, RODA-in provides another useful feature: metadata validation. This feature is only available to metadata files created from templates and the template must have a schema correctly added in the configuration. With the push of a button the tags from the template are temporarily replaced with the corresponding values and the resulting file is validated against the schema. If the file cannot be validated successfully, the application displays the detailed error message and the location of the error, otherwise it only shows a success message.

Documentation can be added to SIPs to complement the information that is being archived in the Data section. Files and folders can be added from the application's file explorer or from the operation system's file explorer using a drag and drop system. Unlike when files are added to the Data section of SIPs, adding documentation does not set the state of the files as mapped, so an archivist can add the same file to the documentation of any number of SIPs. After being added, the files can be removed and rearranged in the same way as in the Data section.

V. Conclusion and Future Work

Our goal was to develop an offline SIP creation tool to tackle the problem of mass data archival. RODA-in allows the archivist to create thousands of SIPs with gigabytes of data in few clicks, without overlooking the needs of more advanced users. Since the file system is the source of the data and the user must have full control over it, we developed a file explorer that supports 3 states: normal, ignored and mapped. The classification scheme can be imported from a previously saved plan or created directly in the application.

To create SIPs there are 4 aggregation rules that should cover the needs of most users: one SIP for each selected file or folder; one SIP containing all selected files and/or folders; one SIP for each file under the selected folder(s); create classification scheme and SIPs based on folder structure. In the same way, metadata can be associated to the created SIPs automatically, using 4 association options: metadata based on a template; same metadata in every SIP; load metadata from data folder; load metadata from a folder. The resulting SIPs can be edited and enriched using metadata forms based on templates, rearranging and removing files from the data section of the SIP or adding documentation. The last step is to export the SIPs to an output folder so that they can be later added to a repository. Currently, RODA-in supports two SIP export formats: BagIt and the E-ARK SIP.

The templating system will be one of our main focus in the future which we will expand to support user variables, auto-generated value definition, default values, possible values enumerations, form field type and order. These are all features that will improve the forms and give the users advanced options to further personalize their templates. The concept of transfer projects is another feature that will be added to RODA-in in the future, where archivists will be able to configure project specific variables that can be used in the templates and preserve some data between executions of the application. Finally, to help the user change metadata across multiple SIPs we plan to add the possibility of batch changes of metadata.

Acknowledgment

This work was co-founded by the E-ARK project, an EC-funded pilot action project in the Competitiveness and Innovation Programme 2007-2013, Grant Agreement no. 620998 under the Policy Support Programme.

References

[1] OAIS2012, CCSDS, "Reference Model for an Open Archival Information System (OAIS)", Magenta book, http://public.ccsds.org/publications/archive/650x0m2.pdf, 2012.

[2] J. Kunze, J. Littman, L. Madden, E. Summers, A. Boyko and B. Vargas, "The BagIt File Packaging Format, 2016, Library of Congress, California Digital Library, George Washington University Libraries, University of Maryland.

[3] PREMIS Editorial Committee, "Data Dictionary for Preservation Metadata: PREMIS version 3.0. Technical Report", 2015, Library of Congress.

[4] Healy Susan, "ISO 15489 Records Management: its development and significance", issn = 0956-5698, Records Management Journal, 2010.

[5] Conseil International des Archives , "ISAD(G): General International Standard Archival Description", isbn=9780969603559, 2000, ICA Standards.

[6] T. Karberg, K. Oolu, P. Randmae, K. Johansen, E. Hougaard and A. Thirifays, B. Domajnko, J. Delve and D. Anderson, ”D3.1 - Report on available best practices”, 2014, E-ARK.

[7] T. Karberg, K. Bredenberg, B. Skog, A. Nielsen, K. Johansen, E. Hougaard, H. Silva, G. Zavrsnik, L. Szilágyi and P. Tømmerholt, “D3.3 - E-ARK SIP Pilot Specification (revision of D3.2, main part of the D3.3”, 2016, E-ARK.

[8] D. Eastlake and P. Jones, “US Secure Hash Algorithm 1 (SHA1)”, 2001, RFC Editor.

[9] R. Rivest, “The MD5 Message-Digest Algorithm”, 1992, RFC Editor, United States.