DAITSS 1.x OVERVIEW

January 2007

© 2006 by the Florida Center for Library Automation.

DAITSS (Dark Archive In The Sunshine State) is a digital preservation repository application developed by the Florida Center for Library Automation. In addition to repository functions of ingest, data management and dissemination, DAITSS supports the preservation functions of format normalization, mass format migration, and migration on request.

DAITSS is a "dark archive" intended to be used as a back-end to other systems, such as digital library applications or institutional repository software. It has no public interface and allows no public access, but it can be used in conjunction with an access system.

DAITSS supports use by multiple customers, whether these are different organizations or different units within an organization. It allows great flexibility in terms of what materials are archived and what preservation treatment is accorded to them.
DAITSS is written in Java and tested on Red Hat Linux (RHEL 4). The data management database is MySQL.

Technical requirements

To run DAITSS, a site will need the following installed. Each application or library is followed by the minimum version number and the license type if known.

- Apache Ant 1.6.5 or higher
- Clam AntiVirus .88 (GPL) or any other anti-virus software that can be called from a command line
- Gzip 1.3.3 (GPL)
- JUnit 3.8.1 (CPL 1.0)
- PJX 1.3.0 (GPL)
- MySQL 4.1.12
- Red Hat Linux
- Sun Java 1.5 or higher

- Sun J2SE 5.x 1.5.0_05
- Sun JavaBeans Activation Framework JAF 1.0.4
- Sun JavaMail API 1.3.3
- Sun rmiregistry
- Xalan-Java 2.6.0 (Apache License 2.0)
- Xerxes2 Java Parser 2.6.2 (Apache Software License 1.1)
- XML Beans 1.0.4

Some other software tools are specific to the treatment of particular formats. These are optional in the sense that they are not needed if a site won't be handling that format, or if the site wants to write its own normalization methods.

- Ghostscript 8.53 (AFPL) to normalize PDF files to page-image TIFF files
- Mencoder 1.0 to normalize compressed video streams in AVI files

Standards compliance

DAITSS implements the functional model defined in *Reference Model for an Open Archival Information System (OAIS)*, ISO 14721:2003.

It uses METS documents as descriptors for Submission Information Packages (SIPs), Archival Information Packages (AIPs), and Dissemination Information Packages (DIPs). (These packages are defined in the OAIS). Requirements for METS descriptors are described in the documentation DAITSS METS SIP Profile.

Internally stored preservation metadata is compliant with the PREMIS Data Dictionary version 1.0. However, not all PREMIS elements are maintained at this time. Full PREMIS support will be added in the future.

DAITSS and OAIS

DAITSS was designed to implement the functional model defined in OAIS. A comparison of OAIS and DAITSS functions is shown below.

OAIS	DAITSS
Ingest functions -- receive SIPs -- perform quality assurance on SIPs -- generate AIPs -- extract descriptive information from the AIP for inclusion in the archive's database -- coordinate updates to Archival Storage	Ingest functions -- receive SIPs -- perform quality assurance on SIPs -- generate AIPs -- extract or generate preservation metadata for inclusion in the AIP and in the archive's database -- coordinate updates to Archival Storage

and Data Management	and Data Management -- create localized, normalized, and migrated versions of files when appropriate
Archival Storage functions -- receive AIPs from Ingest and add them to permanent storage -- perform error checking -- provide AIPs to Access to fulfill orders -- manage the storage hierarchy --refresh the media on which holdings are stored -- provide disaster recovery facilities	Archival Storage functions -- receive AIPs from Ingest and add them to permanent storage -- perform error checking, in the form of ongoing fixity checking and ongoing checking for consistency between the management database and storage -- provide AIPs to Access to fulfill orders (pending) These functions must be provided outside of the DAITSS applications.
Data Management functions -- perform database updates -- perform queries on the archive database to generate result sets -- produce reports from result sets -- maintain schema & view definitions, and referential integrity	Data Management functions -- perform database updates -- perform queries on the archive database to generate result sets -- produce reports from result sets These functions are performed by staff working directly with the relational database
Administration functions -- soliciting and negotiating submission agreements -- auditing submissions to ensure they meet standards -- developing standards and policies -- providing customer support -- many other similar functions are detailed	Administration functions These functions must performed by staff outside of the DAITSS application. DAITSS does allow information from submission agreements to be entered into tables in the system.

In the OAIS model, the Producer (that is, the agent providing the information to be preserved) delivers metadata and content data files together in a Submission Information Package (SIP). In the DAITSS implementation, a SIP consists of metadata in the form of a METS document (the "SIP Descriptor") and content data files.

In OAIS, the archive Ingest function contributes descriptive information to the archive Data Management function, and reformats the SIP into an Archive Information Package

(AIP) for storage. In DAITSS, Ingest populates the management database and reformats the SIP into an AIP. All metadata stored in the data management database is stored redundantly in the AIP.

In OAIS, AIPs are passed to Archival Storage. DAITSS contains a generic storage interface. To use DAITSS with any actual storage system, local programming must be done to implement the generic storage interface with the actual storage system.

In OAIS, the Access function satisfies requests for reports and Dissemination Information Packages (DIP). In DAITSS, an Access function will do the same. This is not yet programmed as of January 2006. DAITSS also has a Withdrawal function, which removes content and associated metadata from the repository, retaining some administrative information.

Object Data Model

DAITSS manages information about digital objects at three levels: the Intellectual Entity, Data File, and Bitstream.

An Intellectual Entity is a coherent set of content that is described and used as a unit, for example, a book, a map, a photograph, an issue of a serial. Each SIP is assumed to contain all the Data Files necessary to render at least one representation of a single Intellectual Entity. (Some necessary files can be automatically added to the SIP by DAITSS.)

A Data File is a single named digital file, such as a PDF, TIFF or XML file.

A Bitstream is a sequence of bits that has common attributes for preservation purposes. One or more bitstreams may be embedded within a Data File. For example, an AVI file contains audio and video bitstreams.

DAITSS maintains relationships between Intellectual Entities, Data Files, and Bitstreams. It associates descriptive, administrative, preservation and technical metadata with the appropriate level of object.

Information Packages

Submission Information Package

Producers are responsible for formatting SIPs appropriately for Ingest. This is not enforced by the system, but each SIP should ideally represent a single intellectual entity (that is, something that can be used and described as a unit, such as a book, a dissertation, a technical report, an oral history, and so on). The SIP must contain a METS document describing the contents of the package (a "SIP descriptor"), and should contain all the data files making up the intellectual entity. (In some cases, DAITSS can supply missing data files.)

Archival Information Package

The AIP contains the original SIP descriptor and all data files from the SIP with a preservation level of "bit" or "full". In addition, it may contain files not included in the SIP, but linked-to from a file in the SIP. For example, if the SIP contains an XML document with a link to an XML schema but does not contain the schema itself, DAITSS will attempt to download the schema and include it in the AIP. In addition, it will contain localized, normalized, and/or migrated versions of files that were created on Ingest. Finally, it contains a METS format AIP descriptor, which includes an XML representation of all of the metadata pertaining to the Intellectual entity, and to each of the Data Files in the AIP.

Dissemination Information Package

The DIP exports the contents of the SIP. It contains all of the files included in the original SIP and any added linked-to files. It also contains the "last, best" version of any file, if different from what was submitted. The METS format DIP descriptor includes all the metadata in the AIP descriptor. It contains one structural map (structMap) section for the original representation and if applicable, a second structural map section describing the current representation.

Preservation strategies

DAITSS supports two levels of preservation, “bit-level” and “full.” Bit-level preservation includes secure storage, data monitoring, and the creation of multiple masters. Full preservation includes bit-level treatment plus localization, format normalization and format migration.

Secure storage

The only access to stored AIPs provided by the DAITSS application is via the Access service which is restricted to authorized users (in general, repository managers). Data cannot be accessed via URL. Controlling access from outside the application is the responsibility of the site administrator. However, because there is no real-time online public access, the application can be firewalled to disallow Internet access.

Data monitoring

Fixity checks are performed on all of the Data Files stored in an AIP by the usual method of calculating a message digest and comparing it to a stored value. DAITSS uses two different digests calculated by different algorithms for each file. Corrupt files are automatically replaced with uncorrupted master copies when possible.

Integrity checks are performed to ensure that all Data Files that are supposed to be in an AIP are actually in storage. The integrity checker uses the management database to obtain the list of all files in the AIP. [

Multiple masters

DAITSS writes "n" master copies of all AIP files, where "n" can be set by the repository manager. In the default configuration, n is 3. (These are considered multiple masters rather than a master and backups because each master copy is independently addressable.) Repository managers can chose whether or not to make backup copies of the masters; this function would be outside of DAITSS.

Localization

If a Data File contains one or more links to other files (for example, an XML file containing a link to a schema definition), a localized version of the Data File will be created on ingest when possible. The linked-to file will be located and downloaded if it is not already contained in the SIP. A localized version of the original, linked-from file will be created in which the link is replaced by a reference to the name of the linked-to file in the AIP. The purpose of localization is to ensure that AIPs are complete and that references between files in the AIP can always be resolved. Localization is performed on Ingest and localized versions are stored in the AIP.

Normalization

If a file is in a format considered to be less than optimal for digital preservation a normalized version of the file may be created. The normalized file contains the same content in a more preservation-worthy format. Normalized versions may not be equivalent to originals in appearance or functionality. For example, a PDF file can be normalized into a set of page-image TIFFs. In this case the appearance of the content is retained but functionality such as actionable hyperlinks is lost. If normalization is part of the action plan for a particular file format, it will be performed on Ingest. The normalized version will not be stored in the AIP, but the entire SIP will be rejected if normalization fails. This ensures that normalization could be done if necessary, but spares the cost of storing normalized versions.

Migration

If a file is in a format considered at risk of obsolescence, a version will be created in a format considered to be a reasonable successor to the original format. The successor format may be a higher version of the original format (for example, PDF 1.4 might be migrated to PDF 1.6) or it may be another format altogether. Migration is performed on Ingest and migrated versions are stored in the AIP. To perform forward migration on a previously ingested AIP, the AIP must be disseminated via the Dissemination function, and (re)delivered to Ingest.

Relationships between all versions of a file (original, localized, migrated, and normalized) are maintained in the management tables.

Customer Information

A DAITSS repository can manage materials submitted by any number of different customers. Customers must establish agreements with the archive indicating the materials to be archived and the preservation level to be accorded them. Each customer agreement corresponds to one account represented by an Account code. Each Account can designate contact persons for administrative issues, technical issues, billing, and reporting.

Each customer must define the desired preservation treatment of materials submitted in an SIP for that Account. This is done by specifying for each digital format (MIME type) either "full," "bit" or "none". For example, a customer can specify that files of type "application/pdf" receive full preservation treatment but "image/png" receive only bit preservation.

In addition, the customer can optionally set up any number of "Project" categories, and specify preservation treatment by format within Project. Project is an arbitrary code which can be used to designate a publication, collection, or any set of materials. For example, the customer could specify that PDF files for project "perm" should get full preservation treatment, and all other PDF files get bit preservation. Or, if the campus Development Office were to archive both a quarterly magazine and a newsletter, these could get different Project codes, so that the magazine could be accorded full preservation treatment and the newsletter only bit-level preservation. Project codes must be pre-defined in DAITSS. They are associated with incoming materials by the PROJECT attribute in the AGREEMENT_INFO element in the SIP descriptor.

The customer may also optionally define any number of "Subaccount" codes. Like Project, Subaccount is an arbitrary code associated with SIPs by an attribute (SUBACCOUNT) in the AGREEMENT_INFO element in the SIP descriptor. Unlike Project, Subaccount has no effect on preservation. Its only function is in billing and reporting, to allow more granular distinctions within the Account. Some anticipated uses of Subaccount are to designate a subunit within the larger customer organization, or to designate an external organization doing its archiving through the customer. For example, the campus Development Office produces a quarterly magazine that is archived by agreement with the university library which is a repository customer. The university library can assign a Subaccount code to the Development Office that is different from the Subaccount code the library uses for its own materials, so that the Development Office can receive its own usage reports

Accounts, Subaccounts and Projects must be predefined in the DAITSS ASAP table.

Finally, each customer must specify an email address associated with the Account for receiving Ingest and Error reports.

Configuration options

In addition to preservation treatment described above, the many options are configurable in DAITSS, including the following.

- How many master copies to write to different storage devices. (default = 3)
- Which message digests to calculate for each file. (default = SHA-1 and MD5)
- What transformation rules to apply to each format on ingest for full preservation treatment; i.e. whether to create normalized and/or migrated versions, and command information for invoking the appropriate routines.
- Database connectivity (JDBC connect string).
- Whether or not to delete packages from the ingest/in/ directory after processing; and whether or not to delete packages from the ingest/out/ directory after writing to storage.
- What severe errors and warnings to include in Ingest reports to customers.
- An email address for repository management to which to blind-copy customer Ingest reports and Error reports.
- The time zone to use in reports and products. (Internally, all times are maintained in GMT.)
- Whether the Preprocessing function should generate descriptors for SIPs that do not have them, and if so, what Account, Subaccount and Project codes to put into the SIP descriptor.

A complete list of configuration options is given in the DAITSS Operations Manual.

Detailed DAITSS Functions by Service

Preprocessing

The Preprocessing, or Prep, function, prepares SIPs for Ingest. It can be skipped if no advanced checking or alteration of packages is desired. Local methods can be added by any repository running DAITSS to edit, normalize or otherwise prepare materials in all or some SIPs. Generic functions of Prep are:

1. Check if the SIP is a single file (at this time .zip is the only aggregation method supported) and if so:

    a. (planned) Check the digital signature (if any) on the .zip file to confirm the source is an authorized depositor and the SIP has not been changed since signed.

    b. Unzip the file.

2. Check that the SIP contains a SIP descriptor. If there is no SIP descriptor, and creation of a SIP descriptor is specified for this Account and Project in the Prep configuration file, then create a METS-format SIP descriptor. If there is no SIP descriptor and none is to be created, reject the package.

3. Check that the SIP contains at least one content file in addition to the SIP descriptor.

4. Delete files in the SIP that are not referenced by the SIP descriptor.

5. Put the resulting SIP in the ingest/in/ directory for Ingest processing.

Ingest

When Ingest starts up, it first makes a list of all SIPs in the ingest/in/ directory. It then performs the following actions for each entry in the SIP list.

1. Check the entry in the SIP list to ensure the package exists and that the package path points to a directory that is writeable and readable.

2. Identify the SIP descriptor and validate it against the METS schema.

3. Extract certain metadata contained in the SIP descriptor for later use. This includes descriptive metadata about the Intellectual Object, and the owning Account, Subaccount, and Project codes.

4. Create a Data File Object for each file in the SIP. This includes the following steps:

    a. Determine the file type by comparing key file characteristics against the characteristics of various file formats, one after the other, until a match is found. The MIME type given for the file in the SIP descriptor (if any) is tried first, followed by the format identified by the file type extension (if any). If no match is found among known formats, the Unix "file" function is used to return a tentative format.

b. Create a Data File Object with a unique persistent identifier for the file. This consists of the following steps:

   - Check the file for viruses. If contaminated, the entire SIP containing the file is rejected.

   - Determine the desired preservation level of the file by looking up the file format and Project in a table of preservation requirements for the Account.

   - Validate the file format against format specifications. Validation may use third-party tools. Any anomalies (points of non-conformance with the profile, such as bad tags in a TIFF header) are noted and recorded. Certain anomalies will cause the preservation level to be downgraded from full to bit level, in which case a EVENT_CHANGED_PRES_DOWN Event will be recorded.

   - Calculate the message digest(s) for the file. Verify any message digest(s) included in the SIP descriptor for the file. In the event of a non-match, data corruption is assumed and the entire SIP is rejected.

   - Use knowledge about the file format to parse the file to find embedded bitstreams, and create a Bitstream Object with a unique persistent identifier for each bitstream.

   - Extract and/or derive technical metadata about the file and its bitstreams.

c. For each file in the SIP, the file is "distributed" (that is, if the format can contain external links) resolve each external link and download the linked-to file. Create a Data File Object for the downloaded file. This can continue iteratively until all links are resolved.

d. For each file in the SIP, if it is designated for full preservation and if there is a forward migration method for its format, create a migrated version of the file and add it to the SIP. Create a Data File Object for the migrated version.

e. For each file in the SIP, if it is designated for full preservation, and if there are references to other files in the SIP, create a localized version of the file replacing references with the full internal file name. Create a Data File Object for the localized version.

f. For each file in the SIP, if it is designated for full preservation, and if there is a normalization method for its format, create a normalized version and add it to the SIP. Create a Data File Object for the normalized version.

g. Create metadata for the Intellectual Entity and for all Data File Objects, including documentation of relationships among them and of events that effected them.

h. Create the AIP. This includes the following steps:

- Identify duplicate files and files with preservation level "none" and do not include them in the AIP.

- Identify duplicates of "global files" (files such as the METS schema which are known to occur in all or many AIPs) and do not include them in the AIP. Replace references to these files with references to the global file.

- Create the AIP descriptor. The descriptor contains a <filesec> with references to all files in the AIP, and complete metadata about all files, events and relationships.

- Write the AIP to the ingest/out/ directory.

- Write n copies of the AIP to storage, depending on configuration options. Depending on format, some files are compressed in a non-proprietary and lossless compression scheme.

- Commit the update to the management database.

i. Delete the SIP from the /in directory.

j. Format ingest information for the AIP as an XML message and email it to the customer and to the archive.

Storage Maintenance

The Storage Maintenance function ensures that stored masters remain good copies on readable media. The "checkers" can be set up to do random or comprehensive monitoring.

1. Fixity checker. For each of "n" stored masters, two message digests are calculated by different algorithms and compared with values in the master database. Corrupt files are replaced with good versions where possible by locating uncorrupted master copies.

2. Integrity checker. (planned) All Data Files referenced in the management database must exist in storage or an error is reported.

Access

The Access function provides a mechanism for billing and reporting. Specific services include:

1. Activity reports. Activity reports are issued when certain actions are completed, such as when a SIP is ingested or an AIP is withdrawn. Activity reports are emailed XML documents. Customers can use stylesheets included in DAITSS or develop their own.

2. Billing (planned)

3. Repository statistics. Total space used in GB, number of Intellectual Entities, number of Data Files and other statistics are available as real-time or batch reports.

4. Customer statistics (planned)

5. Database query (planned

Dissemination

The Dissemination Service exports a DIP to an authorized requestor. The DIP contains the original SIP and the "last, best" version of any file if different from that originally submitted. The AIP remains in storage (it can only be deleted by the Withdrawal service) but because Dissemination includes a re-ingest step, the AIP may be updated during the process.

1. Request authentication. The authentication function is invoked to determine the disseminiation request is authorized.

2. Package assembly. Copies of all files in the AIP are obtained from storage and placed in the re-ingest queue. The dissemination is recorded as an Event.

3. Re-Ingest. The package is ingested again to take advantage of any new format identification or format processing routines that may have been added to DAITSS since the last ingest of this material.

    - Look up the intellectual entity in the database and obtain stored metadata for the intellectual entity.

- For each file in the package, obtain database information for the data file. Parse the data file to extract its metadata and create its bitstreams. If the file type is unsupported, use *ffident* to identify the MIME type. If the format of the file has changed, create a new Data File object and record its metadata. If the data file requires migration, normalization, and/or localization, then create migrated, normalized and/or localized versions as appropriate.

- Identify disposable files (the old AIP descriptor and intermediate migrated files) and mark them obsolete.

- Create a new AIP descriptor. Write the AIP descriptor and any newly created files to storage. Record the ingest as an Event. Delete the obsolete files. Update the management database.

4. Write a DIP. Create a DIP descriptor with separate structure maps for the current and "last, best" representations. Assemble a DIP consisting of the data files in the AIP and the DIP descriptor. Zip up the files in the DIP and create message digests in a digest file. Place the zipped file and the message digest file in the dissemination directory associated with the requestor.

5. Email a Dissemination Report to the appropriate contact address informing the requesting institution that the DIP is available for pick-up.

Withdrawal

The Withdrawal service deletes stored AIPs. It can be used when a customer requests their own content to be removed from the repository, when a customer terminates use of the repository, or when the copyright status of an item is challenged. Management information recorded about the Intellectual Entity is retained, but updated with withdrawal information.

1. Request authentication. The authentication function is invoked to determine the request is authorized.

2. Metadata deletion. Metadata pertaining to all Data Files within the AIP is deleted from the management database. Metadata pertaining to the Intellectual Entity is retained.

3. File deletion. A request is made to Storage Management to delete all master copies of all Data Files in the AIP, including the AIP descriptor.

4. Metadata update. The withdrawal is recorded in the management database as an Event associated with the Intellectual entity.

5. Withdrawal report. Withdrawal information is formatted as an XML message and emailed to the customer and to the archive.

Repository Administration (planned)

The Repository Administration service provides a graphical interface for updating configuration files and profiles needed to run the repository.

Workflows

Workflow for Adding Materials

In a normal workflow for adding new materials to the archive, the customer creates a SIP for each Intellectual Entity to be archived. Optionally, the customer can associate a message digest with each file within the SIP by including them in the METS SIP descriptor. Optionally, the customer can zip the entire SIP and sign it with a digital signature. (planned)

The Ingest Service processes each SIP, updating database tables and storage.

The Reporting Service sends a report to the customer. The report contains DAITSS identifiers assigned to, and message digests computed for, each ingested file, which the customer can import into a local record of files archived.

Workflow for a Forward Migration

Archive management ascertains all files of format A must be migrated to successor format B. An A-2-B migrator is written and added to DAITSS. Configuration files are updated to reflect the addition of the migrator and the formats it will be used for.

Repository management uses the Reporting Service to get a list of all files of format A and the Intellectual Entities of which they are a part.

Repository management uses the Dissemination Service to obtain DIPs for each Intellectual Entity. The DIPs are submitted as SIPs to the Ingest Service, which updates the AIP for each Intellectual Entity, including the new files in format B.