Version 4.10.0 Changes

Version 4.10.0 of SobekCM includes the most comprehensive collection of changes since SobekCM was first released ten years ago. This version continues preparation for the release of version 5.0 by updating the basic architecture of the system to be more configurable and customizable. In addition, the web application is increasing dependent on the discrete microservice architecture provided by the SobekCM engine. This version also adds a new plug-in architecture for adding extensions which add new metadata type, modify or add new item viewers, customize item citation display, add new engine endpoints, add new builder modules, and much more. Over thirty new microservice endpoints were also added to the SobekCM engine as well. The sum of all these changes results in a more open-architecture which can easily be modified and customized for any institution.

In addition, this version adds several new administrative screens for users. A new set of many system-wide settings screens exposes all configuration information used by the system currently, even if they cannot be modified directly through the web interface. In addition, interoperability between the SobekCM Builder service and the web has been increased by adding the ability to view and filter the builder log through the user interface. System administrators can also add or remove builder folders, directly through a new builder folder management screen.

The builder was also updated in this new version. The architecture was modified to take advantage of the new configuration microservices offered by the engine and supports enabled plug-ins as well. The automatic usage collection and monthly emails were updated to make the process run more smoothly and significantly quicker. The builder can also be set to use Tesseract to perform automatic OCR on incoming TIFF files.

In total, nearly one hundred thousand lines of code were changed between the previous version and this version, which represents roughly 35% of the total codebase and 66% percent of the files were changed in some way.

Architecture

The architecture of the SobekCM web application underwent major changes to provide a more open architecture based on microservice endpoints. The core of the system, the display of digital resources, was completely revamped and a new architecture for the item viewers was implemented. While metadata customization has existed in the METS reader and writer for some time, this version completes that process by allowing simple customization of the citation and online forms to allow additional metadata fields to be added fairly simple. In addition, far more portions of the system can be controlled via configuration files, to allow for limitless customization.

In general, the architecture is more focused on pulling the data from the SobekCM engine "just in time", instead of pulling all the data first, and then building the architecture to serve the response in the format requested. This reflects the change that the SobekCM web application will be depended on less to provide JSON and raw data for other tools, since the SobekCM engine endpoints now can provide this functionality through microservices. Collecting all of the data at the very beginning was far less extensible. With the new architecture, a completely different item viewer architecture could be created, using a different base object as well, and be swapped into the system for the current item HTML writer. Allowing changes like this necessitated a greater flexibility in object types and a decrease in the number of limiting enumerations utilized throughout the system.

Configuration

The overall ability to control the look and feel via configuration files was greatly enhanced in this version. During this process, five new default configuration files were added and all the separate configuration readers were consolidated into a single reader. All the sections of the configuration files are now repeatable and many levels of the hierarchy include an ability to clear all pre-existing configuration information. Previously, configuration information for each discrete portion of the system was controlled by a specific configuration file by filename. With this new release, the reader reads all the configuration files and any section of the configuration can be in any of the configuration files. All of the configuration information and the log file generated while reading all the configuration files are stored in the *InstanceWide_Configuration* object and are serializable and accessible through engine microservice endpoints as JSON, JSON-P, XML, and ProtoBuf.

Item Viewer Customization

This version includes major changes to the way item viewers work, although in most cases the resulting display will be very similar to the previous version. These changes normalized the behavior of all the existing item viewers while simultaneously allowing new viewers to be seamlessly added. In addition, the item writer utilizes the new *BriefItemInfo* object API and the new REST microservice endpoints for retrieving item information to display.

Each item viewer contains two classes, the item viewer prototype and the actual item viewer. The prototype is a singleton object that is used to determine if a particular viewer should be included for any digital resource. The prototype also determines if the user has access to the view and indicates how (or if) the viewer should be included on the item main menu. Finally, the item viewer prototype is used by the item view factory to create the item viewer itself which draws the HTML for a single view of a digital resource.

All the assembly and class information for each item viewer prototype is stored in the configuration files. Also in the configuration files is the URL segment that requests the viewer when someone is using the web interface and the code which matches back to the database viewer. The assembly and class to be used when writing the item main menu is included, which allows for even more customization of the main menu if desired. In the configuration information, individual viewers can be marked as management viewers, in which case they are always included in the items. Otherwise, all the views to be included in the resource are pulled from the database and included as a string, rather than a constricting enumeration. All of this makes it much easier to add new viewers to your own instance of SobekCM to fully customize the way the digital resource display works.

The database contains the basic priority of each viewer (i.e., which should be the default viewer) as well as the order they appear on the menu. This can be changed system-wide if desired. In addition, it is now possible in the database to override this and set the viewer order and menu order for a single item or set of items.

The digital resource microservice APIs exposed by the engine, and the data format of the *BriefItemInfo* object was refined and should now be mostly stable moving forward. We are encouraging everyone to work with the item API if possible and adapt its use over any previously used APIs. The *BriefItemInfo* object was modified to include behavior information, geo-spatial data, hierarchical spatial subjects, and other data needed by the user interface. In addition, a structure was added to allow any item-level settings to be loaded from the database

and an extension data structure was added to allow for extensibility of the *BriefItemInfo* object for any plug-ins or extensions.

Metadata Customization

Metadata creation was already very customizable, but several important changes were implemented in this version. A metadata reader/writer or METS section reader/writer can now be referenced from an external assembly in the configuration and the external assembly/class will be loaded. The existing metadata configuration objects were updated to allow configuration to be collected over many configuration files and it now exists in the *SobekCM.Core* library. The settings within the *SobekCM.Resource_Object* library can be set via the new static *ResourceObjectSettings* class. In addition, all the metadata configuration information is serializable so it can be served via the engine architecture.

Citation Customization

Display of the citation can now be customized via individual citation sets defined in the configuration. For each citation element, you can define the metadata term (mapping), display term, search code, Microdata *ItemProp* mapping, and several other behavior flags. Citation set can be specified for individual items through the item behavior screen, allowing the citation to be customized for a single item, or a set of items. The citation information included in emails sent through the system use their own citation set and can like-wise be customized.

For each citation element, you can either use the default citation writer, or you can implement your own citation section writer. Custom citation section writers exist, by default, for the Rights statement (can include creative commons license image and link), Coordinates (based off the GeoSpatial object), and Spatial Coverage (hierarchical, with country, state, county, and city individually linked for searching).

Template Element Customization

The mapping from the template files (which controls the user experience while working with online templates for metadata and behavior editing) was changed to allow for customization and addition of new template elements tied to new (or existing) metadata elements. A new configuration file was added which maps between the template type/subtype values and the individual template element objects to be utilized. The enumeration of types was removed to allow for the addition of new metadata elements. For inter-template element dependencies, a new system was developed where a dictionary is passed around to be written to and read by each template element in the template.

MySobek Subviewers

All individual behaviors for mySobek viewers were moved from the *mySobek_HtmlSubwriter* into each individual mySobek viewer, via behaviors and a newly exposed property, resulting in less hard-coding and more flexibility in the *mySobek_HtmlSubwriter*. In addition, all mySobek viewers are generated by a factory class, to allow for new viewers to be added through configuration in the future.

Admin Subviewers

Similar to the changes to the mySobek viewers, all individual behaviors for administrative viewers were moved from the *Admin_HtmlSubwriter* into each individual administrative viewer, via behaviors and three newly exposed properties. In addition, all admin viewers are generated by a factory class, to allow for new viewers to be added through configuration in the future.

File system customization

This version also introduces a new structure for working with the files within the digital resources. A new *SobekFileSystem* abstraction is used for all file system calls. The standard *PairTreeStructure* file system was also added and is used by default. This new abstraction will allow users to alter the basic way digital resource files are referenced and retained on disk and ultimately should allow SobekCM to work as a front-end to Fedora if desired.

Aggregation Viewer Customization

In preparation for changes to be included with the next version, a new (demo) aggregation viewers configuration file is included. Once implemented, this will drastically alter the way the all the standard collection pages can be displayed. We look forward to including these changes in version 4.11.0.

User Interface Changes

System-Wide Settings

The system-wide settings screen has been completely refactored and now includes both the settings and the configuration information. While the settings stored in the database can be changed by users of high enough permissions, much of the configuration information is display-only. However, all configurations and settings which affect the SobekCM instance can be viewed through the new form.

The settings themselves were greatly altered, with the default help text and categories pushed into the database. Permissions are also pushed into the database. This allows additional settings to be easily added to facilitate any other custom setting needs. Through the database, a completely new settings page can be added to drive any custom modules or plug-ins as required.

The full list of all installed plug-ins can also be viewed from the new settings screen. This allows the administrative details of each plug-in to be viewed and each individual plug-in can be enabled and disabled. Any errors found while reading the extension information is also displayed.

In addition to the changes included above, the following new screens have been added to the system-wide settings:

- Configuration Files
    - Source Files – lists all the configuration files that were read in order
    - Reading Log – log is created as the configuration files are read

- Builder
    - Builder Settings – database settings which impact only the builder
    - Incoming Folders – list of all the incoming folders where the builder looks for bulk loading
    - Builder Modules – list of all builder modules that are currently in the system
- Metadata Configuration
    - Metadata Search Fields – list of all metadata fields at the item level for searching
    - File Readers/Writers – list of all metadata reader/writers included in the system
    - METS Section Readers/Writers – list of all the readers/writers that work on sections within the METS files for metadata
    - METS Writing Profiles – profiles control how the METS files are written by the system
- Engine Configuration
    - Authentication – contains basic information about authentication in the system and all Shibboleth settings
    - Brief Item Mapping – maps between the full METS resource object and the display object
    - Contact Form – controls what fields are available for the default 'contact us' form in the system
    - Engine Server Endpoints – list of all the microservice endpoints exposed by the engine
    - OAI-PMH Protocol – settings related to the OAI-PMH protocol for sharing metadata, including which formats are supported
    - Quality Control Tool – settings related to the division names and behaviors within the online QC tool used to create structural metadata
- UI Configuration
    - Citation Viewer – list of all the citation sets, including details about how each individual citation element is written
    - Microservice Client Endpoints – list of microservice endpoints utilized by the UI web portion
    - Template Elements – list of all the template elements which expose individual metadata elements for editing within the online metadata entry and editing screens
    - HTML Viewers/Subviewers – currently lists all the item viewers and other top-level settings that affect the item writer
- HTML Snippets
    - Missing page – HTML snippet is displayed when a page is requested (by a non-admin) that does not exist
    - No results – HTML snippet is displayed when a search results in no search results

Builder Status (updated screen)

The existing builder status admin screen was overhauled, adding access to the builder logs and the scheduled task information. In addition, access to the old builder log files was removed as these have not been written for a couple years.

The builder log screen allows the logs the builder writes to the database during normal operation to be viewed. The logs can be viewed between two arbitrary dates and can be filtered by BibID/VID to view information about a specific item or title. By default, simple polling entries are omitted, but these can be included as well. This log screen is a great place to keep track of how your builder is running!

In addition, the builders's scheduled tasks can be viewed in a new tab.

Builder Folder Management (new screen)

A new builder folder management screen was added which allows new builder folders to be added to the system and also allows existing builder folders to be edited. In addition, basic behavior flags can be set to control the types of incoming submission packages accepted. This new screen is available from the system-wide settings under the builder options.

Item Behavior Editing

From the item behavior screen, you can now select the citation set to use. I addition, the viewer selection portion has been modified. The way item viewers work now, many viewers are automatically added to each new item. So, every viewer marked in the database to be added are added by default. Many of the viewers will not appear though, unless the appropriate file type or metadata exists. For example, the Google Map viewer will be added to all items by default, but will only manifest itself if there is coordinate data in the record. The PDF viewer will also appear on all items when editing the viewers in the database, but will only manifest itself if there are one or more PDFs to display. You can now suppress the PDF viewer, even if a PDF is present, by removing it on the item behavior screens. Attributes and files no longer appear on the item viewer selection element either.

Plug-Ins

A new extension architecture was added to allow for additional assemblies of custom code and configuration changes to be created within plug-ins that can be shared with the larger community. A properly constructed plug-in can be dropped into a plug-ins subfolder under the web server and then enabled through the system-wide settings administrative screens. The plug-in can reference DLLs of custom code and include any number of configuration files to make changes to the basic behavior of the system. Plugins contain administrative information such as the version, author, description, and basic plugin name.

A sample plug-in is included in this version release and the full source code of the plug-in will be released as open source shortly. Additional training materials and videos will be released shortly to provide instruction on creating your own plug-ins.

Builder

The builder architecture and configuration has also changed with this version. The new configuration file should contain URLs for engine endpoints which the builder will use to pull configuration information from the engine. This was necessary to allow the builder to see all the configuration information which was collected from all the configuration files and plug-ins. In addition, the builder will copy any assemblies that are part of an enabled plug-in to allow a local copy under the builder executable. With these changes, the builder now fully supports the plug-in architecture as well.

Logging was tweaked and several unused settings were removed. Likewise, the old log files will be deleted except for the last 30 days' worth each time the builder runs. The main builder logging is now in the database

and the inconsequential database logs are expired after 30 days. These are all the log entries indicating no work was completed but all the builder folders were polled.

The builder no longer pulls the entire list of items in the repository each time it checks for new items, and will do a case-by-case check for each incoming item now. This makes the system more performant, use less memory, and less database resources continually.

A new builder module was added to allow incoming TIFF files to be OCRd using an installed version of Tesseract OCR software as well. This new module will be undergoing extensive testing over the next month and we will likely be adding a level of configuration to support different languages and scripts over differing resource types.

The architecture to support scheduled tasks was updated as well with new procedures and data being added into the database. A description was also added for each builder scheduled task and the scheduled tasks can be viewed in the system-wide settings.

Finally, the automatic usage statistics process was updated to handle exceptions better and to run considerably faster. Exceptions within a single log file are handled more cleanly; the log file is skipped and processing continues to the next file while logging the error. Several values were occasionally unexpectedly null during the processing, but these nulls are now handled gracefully as well. Finally, the usage statistics module utilizes a dictionary to lookup the bib/vid when combining individual daily data, rather than using a datatable select, which greatly increases the speed the process runs for large repositories.

Engine

Basic Architecture

The engine architecture was updated to add new protocol formats and to decrease the memory use. The process which invokes endpoint methods was modified to reduce the number instances of the endpoint classes in memory, resulting in more object. The local web server can now always access all of the engine endpoints. Endpoints can support *text* as a protocol, returning plain text for a microservice request. This protocol can be used, for example, when viewing the configuration reading logs.

A *direct* protocol was added for instances where the engine and UI are running in the same context, but was not heavily tested. The direct protocol would take advantage of the shared memory to avoid the cost of serialization and deserialization. Anyone who uses this protocol should include a fallback to a more traditional protocol though. In the case where the memory was completely consumed, an object could potentially be eradicated from memory as soon as it was added, which could cause a race condition.

Endpoints and Serialization

The number of microservice endpoints served by the engine doubled with this version. Many endpoints were added to support item-level data as well as the configuration and setting information. Each of these endpoints required making the objects serializable and present in the *SobekCM.Core* library. For each object, the serialization was reviewed in each format and optimized for smaller package sizes and more normalized XML.

The following new endpoints were added for the top-level configuration, setting, and status information:

- admin
    - settings - Gets the administrative setting values, which includes display information along with the current value and key
- plugins
    - disable - Disable an existing extension/plug-in
    - enable - Enable an existing extension/plug-in
- builder
    - folder - Gets the information about a single incoming builder folder
    - logs - Gets builder logs for a specific date range or filter
    - settings - Gets the fully built builder settings object, including the incoming folders and builder modules
    - status - Gets the latest update on the builder status, including the relevant builder setting values and updates on the scheduled tasks
- config
    - authentication - Gets the authentication configuration information from the configuration files
    - briefitemmapping - Gets the brief item mapping configuration information from the configuration files
    - citation - Gets the citation (UI) configuration information from the configuration files
    - complete - Gets the complete (non-UI) configuration information, read from the configuration files
    - contactform - Gets the contact form configuration information from the configuration files
    - engine - Gets the engine endpoints configuration information from the configuration files
    - extensions - Gets the extensions endpoints configuration information from the configuration files
    - log - Get the log information from the reading of all the configuration files
    - mapeditor - Gets the map editor configuration information from the configuration files
    - metadata - Gets the metadata reading and writing configuration information from the configuration files
    - oaipmh - Gets the OAI-PMH configuration information from the configuration files
    - qctool - Gets the Quality Control tool configuration information from the configuration files
    - static-resources - Gets the static resources (mostly URLs for javascript, images, css, etc..) for the user interface
    - templateelements - Gets the template elements (UI) configuration information from the configuration files
    - ui - Gets the user-interface configuration information from the configuration files
    - viewers - Gets the viewers (UI) configuration information from the configuration files

The following item-level endpoints were added:

- items
    - bytitle - Gets the list of items within a single title
    - cache/reset - Clears information about a single digital resource from the cache
    - ead - Gets any special EAD information related to a digital resource
    - files - Gets the list of files related to a digital resource
    - internal - Gets the brief item information with internal information
    - marc - Gets the marc record transfer object for a bibid / vid
    - tracking - Gets the workflow history and milestones for a digital resource
    - usage - Gets the month-by-month online usage history for a digital resource

Miscellaneous

In addition to the major changes above, many small fixes were included in this version. Many of these fixes were reported by users through the SobekCM ticket system. Below is the full list of minor changes.

Metadata

1. Added a new EVENT note type which maps to/from the MARC field 583.
2. Added the generic XML reader classes, used for reading any structured XML (like TEI) with a mapping set.
3. Added a new bibliographic mapping structure that does not use enumerations, making the mapping more extensible and open.
4. Updates for interoperability with DSpace output METS
    a. Language support
        i. Added ISO 639-1 code mapping
        ii. RFC codes (as text) detected and mapped to language name and correct RFC code
    b. Added a new PROVENANCE note type
    c. METS reader now handles the PARENT type structMap, with mptr tag, and type of HANDLE. In that case the href is mapped into the aggregations behavior field
5. Metadata browses corrected (SOBEK-94)
6. Citation view corrections
    a. VRA Core measurements now includes the units (SOBEK-93)
    b. IEEE-LOM Difficultly label corrected (SOBEK-92)

Utility

1. *SobekCM_METS_Finder* now takes base directory and bibid/vid and returns directory or mets file names
2. Also, updated the *SobekCM.Resource_Object* library to work correctly for ad hoc applications
3. More of the application cache objects can be SET for use in external tools w/o database connectivity
4. The database portion of the *SobekCM.Resource_Object* library was pulled out into its own library so the resource object has less dependencies for general use.
5. Added SQL code to change the parent of a collection

Web Content Pages

1. Corrected issue with building hierarchy of web content pages
2. Web content hierarchy should now clear when settings are requested to be reset
3. Fix for issue saving new web content pages
4. Deleting a web content page corrected and if the web content source page is missing, can still recover and delete (WOLF-36)
5. Fixed the way the web content hierarchy serializes (and deserializes) from XML
6. Option to inherit style from the parent when adding a new page now always works
7. Deleting a web content page resets caches correctly and is immediately reflected (barring browser caching of course)

Other

1. 'sobekcm' removed from reserved keywords
2. Added ability to set where main menu should be written in custom home pages
3. Update to random item stored procedure to include fix to exclude items without pages
4. Updated query string analyzer to better handle exceptions with page requested in item over 65,536.
5. Corrected IP restriction algorithm to handle two IP ranges starting with the same IP without an error
6. SQL script corrects OAI-PMH that allowed private items to be harvested
7. After doing an exact search (like from Browse By or citation links) facets did not work correctly (WOLF-38)
8. Added a presentation table around item viewer so that navigation bar and main area don't conflict and push main area below nav bar (SOBEK-146). This corrects viewing a JPEG2000 in a reduced window pushing it down below screen "fold"
9. Confirmed the identifier facet is working when BrowseBy works correctly (SOBEK-125)
10. Changed the default behavior of the GO button for collections that have browsing all items disabled (SOBEK-115)
11. Fixed problem with the internal aggregation tree view not linking correctly (SOBEK-99)
12. Fixed issue with the magnifying glass not working on aggregation element in edit behaviors view (SOBEK-97)
13. Restored footer to item viewer default behavior when viewing JPEG2000 (SOBEK-96)
14. For non-admins, inactive collections and zero stats collections hidden in all statistics views (SOBEK-90)
15. Corrected issue with the calendar pop-up buttons not appearing correctly (SOBEK-88)
16. Also added images locally in case someone chooses not to use the CDN
17. Mass update works again (couldn't find template file since recent change) (SOBEK-83)
18. METS writer now excludes pages without any files attached (SOBEK-82)
19. Changes to citation viewer to fix how Notes display label is used
20. Fixed javascript for source and holding to not add the pipe value at the beginning of the value when a code is selected
21. Corrected issue when new items were added online the full citation information was not being built properly (and metadata not saving properly?) until edited again
22. An empty facets value in the aggregation XML configuration file does not cause crash anymore
23. Reworked the map editor configuration reader significantly for major performance gains
24. Update changes the way static resources are referenced and read from the configuration files and added a new static *Static_Resources_Gateway* object
25. If there is an error creating derivatives within the builder, then an email is sent to the system admin (SOBEK-228)
26. SessionID and UserID can now be included in the header and footer for javascript purposes (SOBEK-202 USF)