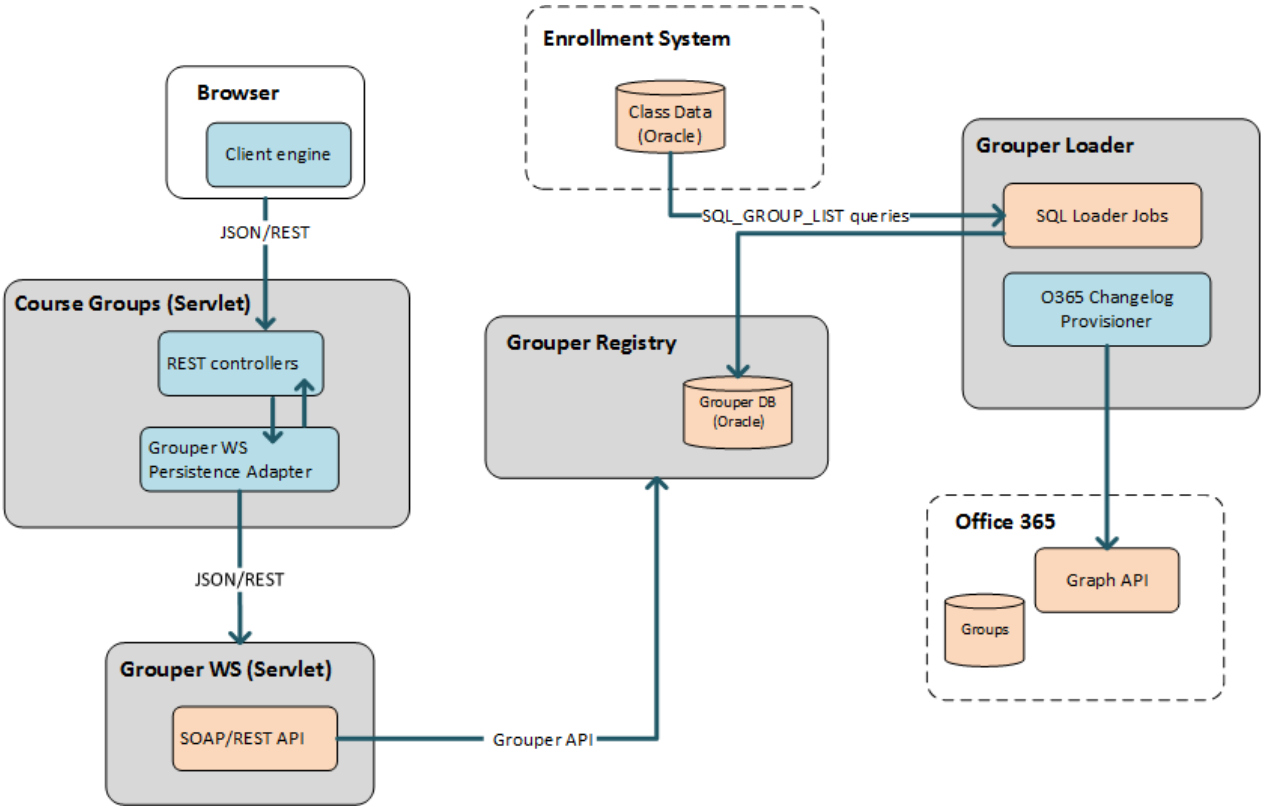
University of North Carolina - Course Groups for Office 365

- Source Data
- Web Application
- Provisioning to Office 365

Completed in March 2018 as a pilot project, the Office 365 Course Groups application is an integration of multiple technologies, providing a simple way for class instructors to set up Office 365 groups from their class roster. Instructors can log into a web application to view their assigned course sections for the current term, or for future terms. They can choose from among the sections for a single course, creating an O365 group out of the combined membership of those sections. These groups can be optionally synchronize changes in enrollment from the system of record. Grouper's many components play a central role in tying the system together.



Course Groups software architecture (blue indicates customized components)

Source Data

In Grouper, a new tree was created to store enrollment data from our HR system. Enrollments are maintained for a one-year rolling window: the current term, previous term, and two future terms. The application divides the enrollment into various roles; instructors can create groups, managers can create groups on behalf of instructors, teaching assistants can optionally be included as either O365 group owners or members, and students make up the bulk of the O365 group membership. Because the application requires data at the level of individual course sections, groups and folders needed to include that level of detail. The folder structure in Grouper thus has the structure:

- unc:base:course
  - fa17 (FA17 - 2017 Fall)
  - sp18 (SP18 - 2018 Spring)

- comp (COMP)
  - 110 (110 - INTRO PROGRAMMING)
    - 001 (COMP110.001.SP18)
      - assistant
      - instructor
      - manager
      - student
- s118 (S118 - 2018 Summer I)
- s218 (S218 - 2018 Summer 2)
- etc
  - security
    - courseReaders (Course Readers)

A number of Grouper Loader jobs were created to import the source data into the folder. One SQL_GROUP_LIST job was created for each role*, and each was schedule around the same time but with different priorities.

- etc
  - loader
    - courses
      - base_group_assistant_loader
      - base_group_instructor_loader
      - base_group_manager_loader
      - base_group_student_loader

**It was almost possible to combine the instructor, assistant, and manager queries into a single job, since they pulled from the same source table. However, the job attribute "GroupsLike" interfered with this option. This is a SQL "%" wildcard pattern specifying which groups are in scope for the query. The value "unc:base:course:%" couldn't be used; the loader query didn't return student groups, so it saw all the existing student groups (created from the previous student loader job) as invalid and deleted them. This lead to a repeating cycle of creating and deleting groups. If the GroupsLike parameter allowed for multi-valued patterns or for more complex patterns, we could have used this approach.*

The import of this new data source significantly increased the number of groups and memberships in Grouper.

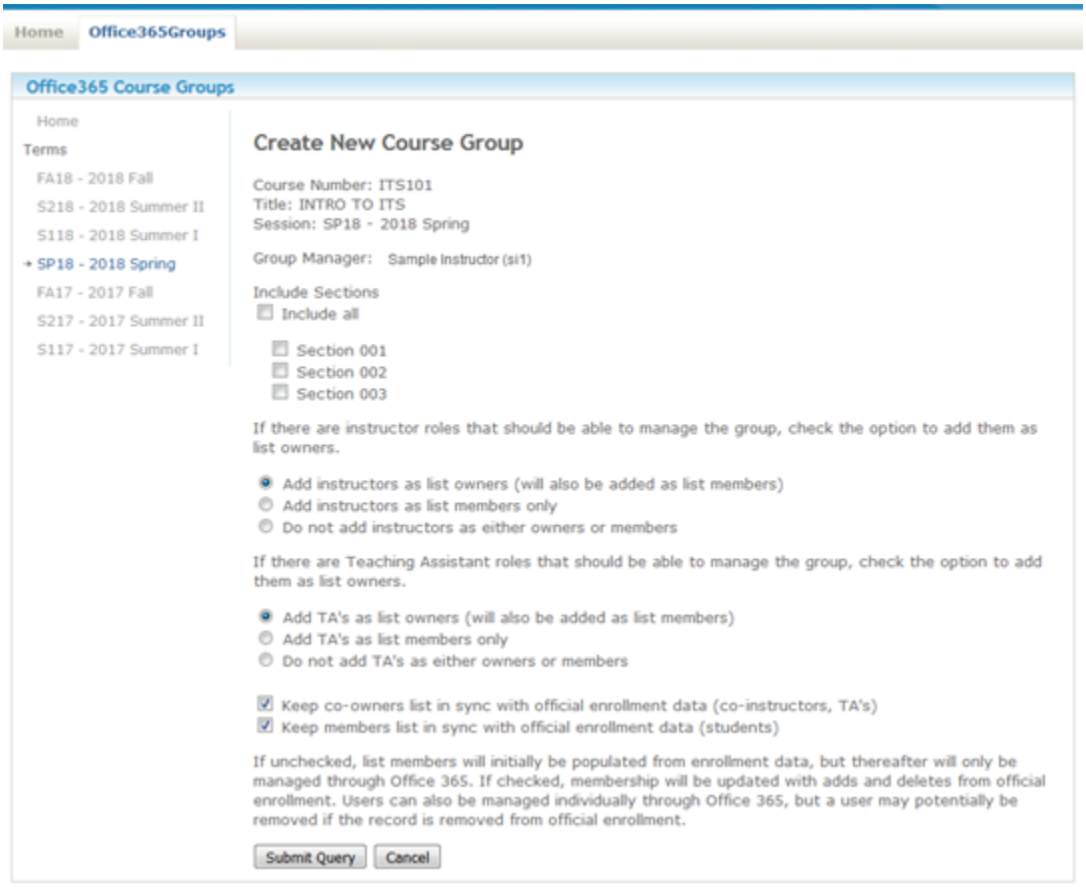
	Before	**After**
Stems	3,200	55,000
Groups	19,000	166,000
Immediate Memberships	239,000	630,000

The number of entity records, audit logs and changelog entries that the initial load created put a temporary strain on our database. In non-production environments, our Oracle redo log space needed to be increased to 40 GB in order to completely load the initial set of groups from the loader jobs (in production, the space was already much higher, and there were no issues with the load).

The Loader jobs are scheduled to run three times per day in production. Each of the four Loader jobs individually take 10-15 minutes to complete with changes to existing groups, or 30-45 minutes on an initial load or addition of a new semester to the data. The SQL queries all run in under 30 seconds; the bulk of the operations are in Grouper comparing and updating the memberships for the large number of groups involved.

Web Application

The web application presents the logged-in user with a list of the courses and sections for which he is an instructor or manager for, for the currently selected term. For a given course, the application queries the groups under unc:base:course:<term>:%:instructor (and the corresponding manager groups) to return a list of all sections having the user as an instructor/manager. The user selects one or more sections from a selected course, chooses other options -- whether to add TA's as members or owners, and whether to sync the enrollment after the initial creation -- and submits to create a new O365 group.



During the design phase of this project, the decision was made to leverage Grouper's Web Services component for database operations, rather than using Grouper API methods which would require direct database access. This made the security architecture simpler, as no special firewall settings were needed, and no additional database access was needed in order to develop the application. The stateless nature of the REST methods required careful consideration of performance issues and security strategies. For example, SQL join operations were generally not available, so workarounds such as multiple queries and locally cached data structures were sometimes needed. We utilized the Ehcache library for caching of certain requests and objects to improve performance where possible.

Provisioning to Office 365

When an O365 group creation is submitted online, a new folder and set of groups is created in Grouper. The folder contains the new name for the group (This is based on the course plus an incrementing number), and under this folder are a members group and an owner group. These are created under the Application tree, for example:

- unc:app:its:o365:course:
  - sp18:
    - biol:
      - 495:
        - biol495_sp18_gp01
          - members
          - owners

The Owners group will contain the user who created the group as a direct member. The Members group will contain the students groups from all of the base sections the user chose to include. The corresponding base instructor and assistant groups may be included in either the Owners or Members group, depending on the user's options, or optionally excluded. By using the base enrollment groups in the membership of the Members and Owners groups, the contents will be kept in sync as the Grouper Loader jobs update the source data.

The Office 365 provisioner is a Grouper changelog consumer based on Unicon's Azure AD provisioner. The modifications we made to the original project were to support our specific needs, and include:

- Allow an HTTP proxy configuration (our Grouper Loader server did not allow direct outgoing connections);
- Add more configuration parameters: subject source, subject attribute, AD domain, group name EL expression, display name EL expression (the EL expressions were used to craft environment specific names, as we had multiple non-production environments targeting the same testing O365 tenant);
- The changelog consumer syncs either the O365 members or owners, depending on the group name;
- The O365 group name is based on the parent folder of the group;
- Removal of the sync attribute flag does not trigger deprovisioning of the O365 group, but rather removes the sync capability (this is controlled by the user through the web interface)
- Deletion of the group does not necessarily trigger deprovisioning form O365 (configured through a custom attribute)
- Adding the sync attribute flag performs a full sync with O365, in case the attribute was added to an existing group;
- Upon group creation, along with setting an attribute for the O365 group guid, the consumer also stores the tenantId and the group name, which can be used to construct URL links to the group (outside of Sharepoint, the group's guid is surprisingly not utilized in web links).
- Optionally set visibility of created groups to private

The Grouper Loader configuration contains the original parameters from the Unicon solution, merged with our extensions. We changed the name of the consumer from "o365" to "o365unified", to avoid a name conflict in case both the original Unicon group-based provisioner were ever used. Examples of the parameters found in grouper-loader.properties are:

(Original)

- changeLog.consumer.o365unified.class = edu.unc.its.idm.grouper.changeLog.consumer.Office365ChangeLogConsumer
- changeLog.consumer.o365unified.quartzCron = 0/30 * * * * ?
- changeLog.consumer.o365unified.syncAttributeName = etc:attribute:office365:o365Sync
- changeLog.consumer.o365unified.retryOnError = true
- changeLog.consumer.o365unified.clientId = e40ad6b4-d00d-4fc5-941c-a1b82bf5c8d
- changeLog.consumer.o365unified.clientSecret = NDUyNDYxY2E2NzI1YjU2NGZiYWEwNmZiYmFiN=
- changeLog.consumer.o365unified.tenantId = testexamplecom.onmicrosoft.com
- #(using default) changeLog.consumer.o365unified.scope = https://graph.microsoft.com/.default

(Custom)

- changeLog.consumer.o365unified.subjectSource = pid
  - Use only this subject source for lookups to improve performance
- changeLog.consumer.o365unified.subjectAttribute = uid
  - the subject attribute to use for the principal in O365
- changeLog.consumer.o365unified.adDomain = adtest.example.com
  - the domain of the principal in O365 (e.g. the full principal is abc@adtest.example.com)
- changeLog.consumer.o365unified.groupNameEL = grouper-TEST-${stemExtension}
- changeLog.consumer.o365unified.groupDisplayNameEL = (Grouper TEST) ${stemDisplayExtension}
- changeLog.consumer.o365unified.groupDescriptionEL = ${ stemDescription ? stemDescription : "Grouper path: " + stemPath }
  - The group name and other properties can be computed by an EL template, using variables:
    - group object
    - parent stem object
    - parent stem full path
    - parent stem extension
    - parent stem display extension
    - - parent stem description
- changeLog.consumer.o365unified.proxyHost = proxy.example.com
- changeLog.consumer.o365unified.proxyPort = 80
- changeLog.consumer.o365unified.removeDeletedGroups = false
- changeLog.consumer.o365unified.visibility = private

New attributes were created to control the provisioning behavior. Since the Grouper folder represents the overall group settings for Office 365, these attributes were saved at the folder level:

- etc:attribute:office365:o365GroupId 778d39f6-4532-4271-8216-c67f0b09c9b
- etc:attribute:office365:o365GroupName grouper-TEST-its101_sp18_gp03

- etc:attribute:office365:o365GroupTenantId example.onmicrosoft.com

Two other attributes applied at the level of the Members or Owners group:

- etc:attribute:unc:o365GroupCreator 702389999
- etc:attribute:office365:o365Sync (marker) true

The o365GroupCreator attribute is used by the web application. For the groups created by an instructor, only the creator can access the created group to change its values. For groups created by a manager on behalf of an instructor, the assigned instructor has the access to edit the group.

The 0365Sync attribute is the Grouper Loader parameter indicating whether changelog entries apply to the O365 changelog consumer. When creating the group through the online form, the user can choose whether to sync future changes, or to create the group with an initial membership based on the current snapshot but ignore future changes in Grouper memberships. If the user chooses not to sync, the application will initially set the o365Sync and then immediately unset it. The initial setting will add a changelog entry that triggers the initial full sync. Because the changelog consumer was modified to handle the attribute deletion as a null operation, the subsequent unsetting will have no effect except to remove the group as a candidate for future changelog actions. The application web page for the group also has a button to force an immediate full sync of the group with O365. If the group is already set to synchronize (but may be inaccurate for some reason), the action unsets and then resets the attribute, to trigger a full sync. If the group is not set to synchronize, the action sets and immediately unsets the attribute, which also forces a full sync.