# University of Washington's migration to Grouper

| Wiki Home | Grouper Release Announcements | Grouper Guides | Grouper Deployment Guide | Community Contributions | Internal Developer Resources |
|---|---|---|---|---|---|

This is a summary of the University's experience migrating from an LDAP group database to Grouper, in the spring and summer of 2010. Our desire, throughout the migration, is that it would be completely seamless — no user or client activity would be interrupted at any time. We were able to achieve this goal.

## Existing groups web service maintained

UW had an existing group web service (GWS) and RESTful API that included a browser UI implemented entirely in javascript---using the GWS API as its resource. We wanted to keep this. We also had groups in LDAP directories. We wanted to keep these as well.

Our old web service was a fastcgi application that used a local templating system and the LDAP directory itself as the group registry. We rewrote the web service as a Tomcat, Spring and Velocity application, that uses the Grouper API to access the a Grouper registry. This seemed a natural fit from the start and has turned out pretty well.

There are two resources that absolutely must be available 24 hours a day and 7 days a week: effective groups for a user, and effective isMemberOf for a group and user. These are needed for most logins to the University's online services. These requirements imply:

1. responses be quick (say, quarter second or less) and
2. the service not have a single point of failure.

To accomplish these we resource both of those queries from the LDAP directories, instead of from the Grouper registry. This implies in turn that the LDAP directories are always up to date.

## Subject database unaffected

We had an existing LDAP directory of subjects that worked pretty well with Grouper. Due to the distributed LDAP source adapter's intolerable habit of reconnecting on all queries, we were led to write a modified version that maintained persistent connections.

We had some subjects not in LDAP, DNS hostnames and federation ePPNs. These we implemented as RDBMS subject sources.

## LDAP directory updates

The distributed tool for keeping an LDAP directory up to date did not meet our performance requirements. Fortunately we had some existing code, from the previous web service, that updated our directories quite efficiently. It was not too hard to re-purpose that to become an efficient directory updater, working from Grouper's change logs. As a result, our LDAP directories are only a few seconds behind the registry itself.

## Automatic provisioning and reconciliation

Most of our groups are provisioned from automatic data feeds: faculty, staff, student affiliations; courses and course rosters; department assignments; and some others. The RESTful model used by our GWS makes this provisioning very easy. From each source groups are created and PUT to the web service.

Our preexisting LDAP group registry is one such source. Each day (or hour, toward the end of migration) we scan an ldif of the registry and PUT each group to the new service. The first time we ran this process it initialized the Grouper registry with all the LDAP directory groups and memberships. Afterwards 99% of the PUTs amounted to effective no-ops at the web service, as none of the information had changed.

- See this example reconciler using the Grouper Client.

## Migration

Our migration plan was simplicity itself:

1. Bring up the new service and let it run in parallel with the old.
2. Migrate automated provisioning to the new system.
3. At this point our clients can use either interface.
4. Switch the routing of all traffic to the new service.
5. Done. (And even people with active browser sessions never noticed the migraton.)

The difficulty of the parallel approach is that synchronicity must be maintained between the old and new registries, while both are accepting updates from their web interfaces. We already had the grouper registry to LDAP tool, which processes updates immediately. We also had the LDAP (as an ldif) provisioner, which processes updates hourly or daily. So the components for synchronization were all present.

When we switched all users to the new service the LDAP reconciliation channel was closed.