Running List: Comanage Stuff Framework – Services - Appliance

Next Steps - Appliance

Next Steps - Service

- Internet2 to put together a service instance in skunkworks, using previous code, Shib, Grouper, etc.
- Dutch to do the same, using…

Some Next Steps- Marketing/PR

- Graphics/slides for comanage as a lightweight collab platform linked to the waffles of federation… - Surfnet
- Graphics/slides for comanage refactored for enterprise or federation level deployment
- Video of the four types of users using comanage/apps
  - Ala http://www.jisc.ac.uk/whatwedo/themes/accessmanagement/federation/animation.aspx

Next Steps - Process

- Figure out groups – dev, -community, -??
- Add Leif, Ian, Thomas to right groups
- Figure out communication modes and times for each group

Next Steps- Framework

- Understand scope of framework – what concepts included/excluded
- Understand, for concepts within scope, the specifications linked to the concept
  - Specific instance – domestication? Is the externalization LDAP or SAML or…
- Application service registry issues (coordination of domestication)

More steps- framework

- Identify good candidates/work-arounds for the missing elements of the model, e.g. STS, provisioning
- Open up an account linking discussion

Current Comanage Materials

Positioning COmanage

- Comanage is not intended as an enterprise-class approach, though many enterprises and federations may well deploy large numbers of instances or a “refactored for industrial use” implementation
- Comanage is intended as a collaboration-class approach that works well and sustainably with enterprise, federated and interfederated infrastructure
- Collaboration-class means lightweight in scope of services commonly managed (just IdM), minimal application requirements, easy implementation options (for example as a collaboration support appliance offered in a cloud), lack of enterprise oriented features (such as a full ESB), etc.
- Works well and sustainably with enterprise, federated and interfederated infrastructure means that Comanage can easily and gracefully link Comanage and federated accounts, work with data feeds from enterprise services, be refactored to leverage different types of infrastructure, etc.
- A lightweight collaboration support approach that integrates with deeper infrastructure

Four Types of Folks

- Sysadmin – installs apps and comanage
- Collabmin – the primary collaboration flywheel; a “steveo”
- Power User – e.g. a PI who wants to be able to do some basic commands (e.g. add users to groups) themselves
- End-user/collaborator – goes directly to domesticated apps or maybe a VO dashboard

STS services

- {K, SAML} in, GridShib cert out
- Pubcookie in, SAML out
- Authn in, dedicated user/pwd out
- SAML token in, webcookie out

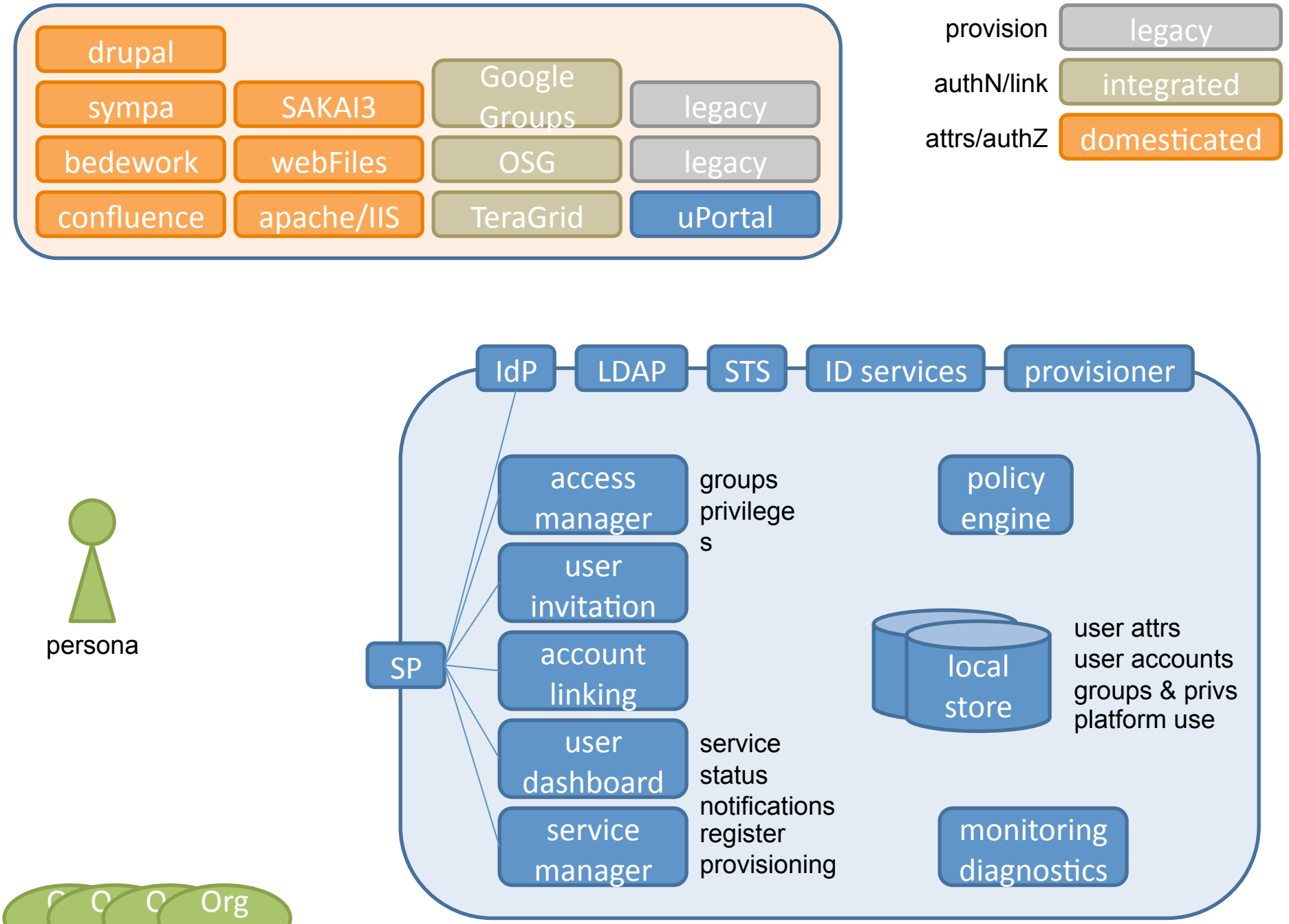
- Binding the framework to app development environments:
  - At what level does stuff need to be specified
  - Which development environments
    .NET, php, Apache
  - Who will write the services

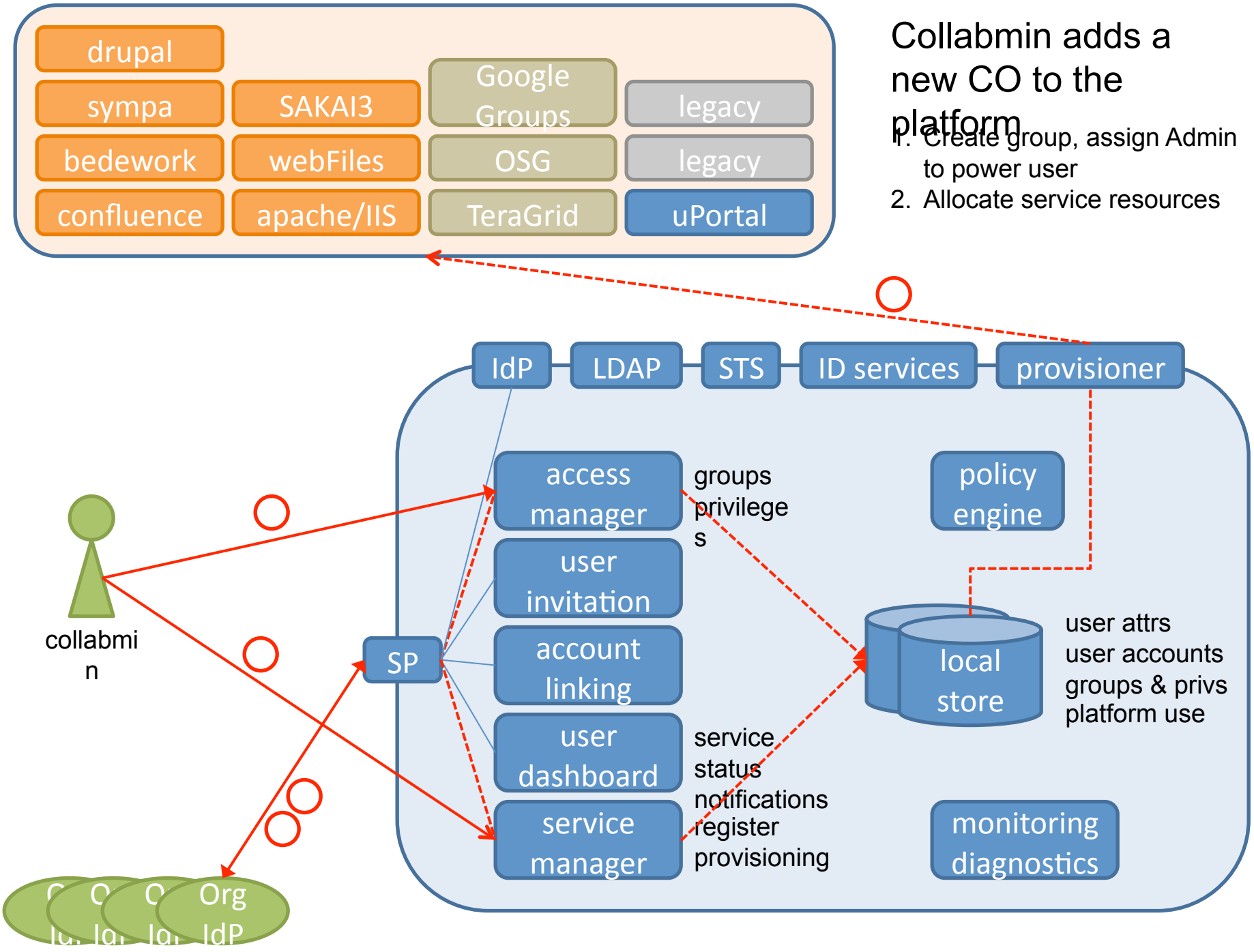
Framework -1

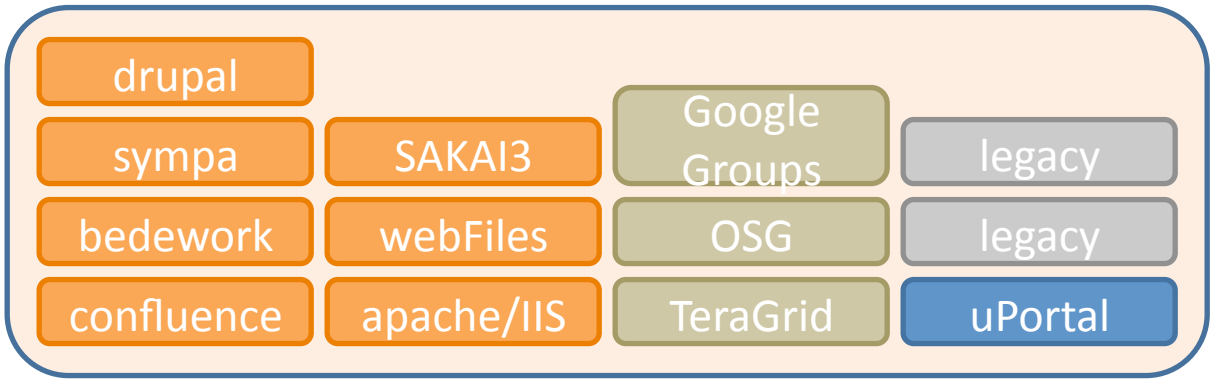
- Several different but consistent perspectives, for different audiences –
  - CIO (block functionality flows)
  - Apps developer – (API's, services, etc)
  - User (user workflows, for different types of users)
  - Others?
- Framework also has layers
  - Language and tech specs
  - Data and metadata specs (to follow later)
  - Others?

Block flow framework parts

- A local datastore
- STS (security token service, aka credential convertor)
- Provisioning/deprovisioning into local store service
- An account linking mechanism
- Group and privilege manager (represent as unified for now)
- SP stub
- Local IdP
- Invitation engine
- Plug and play service for apps that want it
- Attribute services (?)
- Policy engine
- System monitoring and diagnostics
- User dashboard that includes a user collaboration data feed service

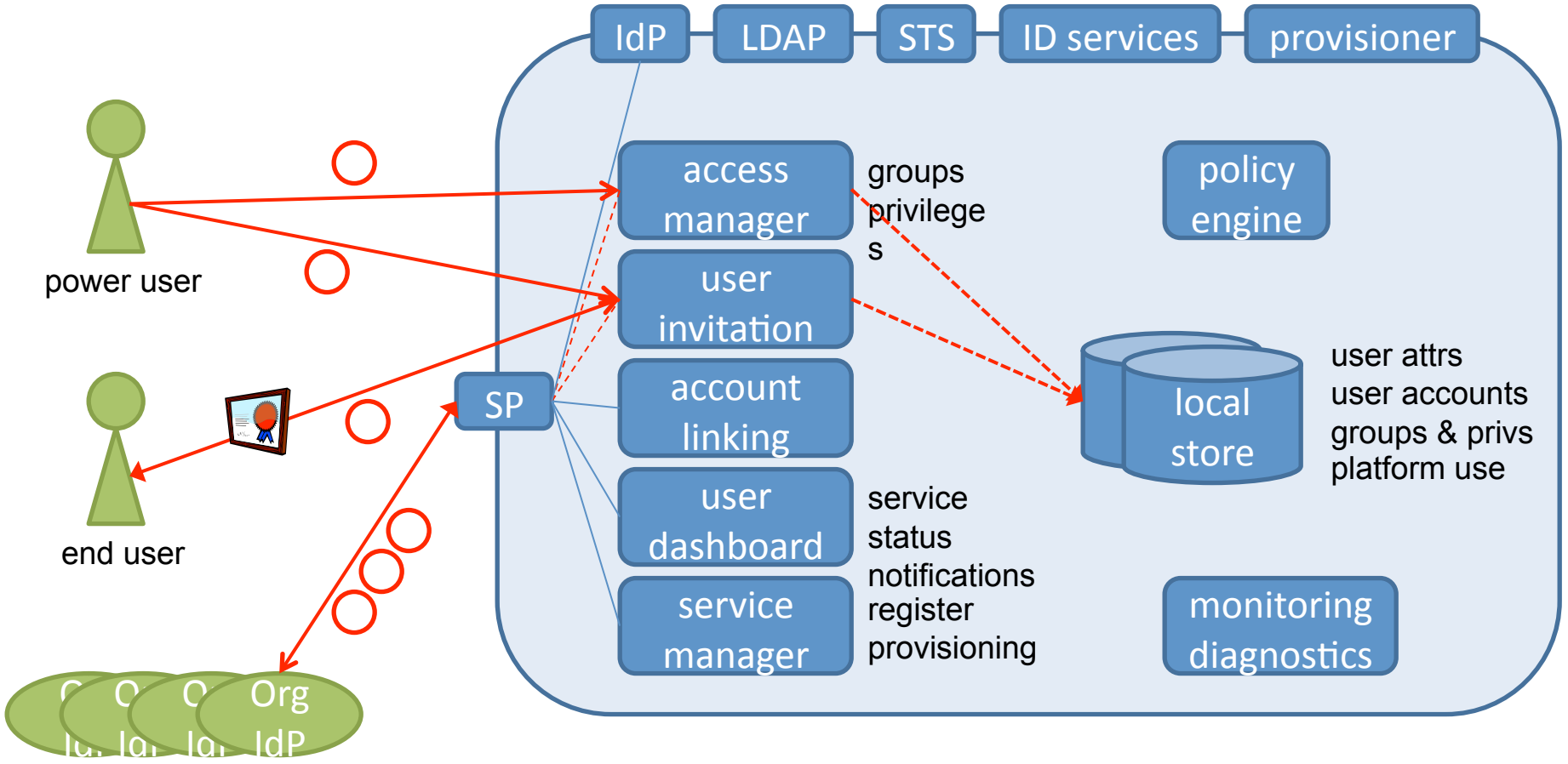

drupal
sympa
SAKAI3
Google Groups
legacy
bedework
webFiles
OSG
legacy
confluence
apache/IIS
TeraGrid
uPortal
provision
legacy
authN/link
integrated
attrs/authZ
domesticated
IdP
LDAP
STS
ID services
provisioner
persona
SP
access manager
groups
privileges
user invitation
account linking
user dashboard
service status
notifications
service manager
register
provisioning
policy engine
local store
user attrs
user accounts
groups & privs
platform use
monitoring
diagnostics
Org IdP

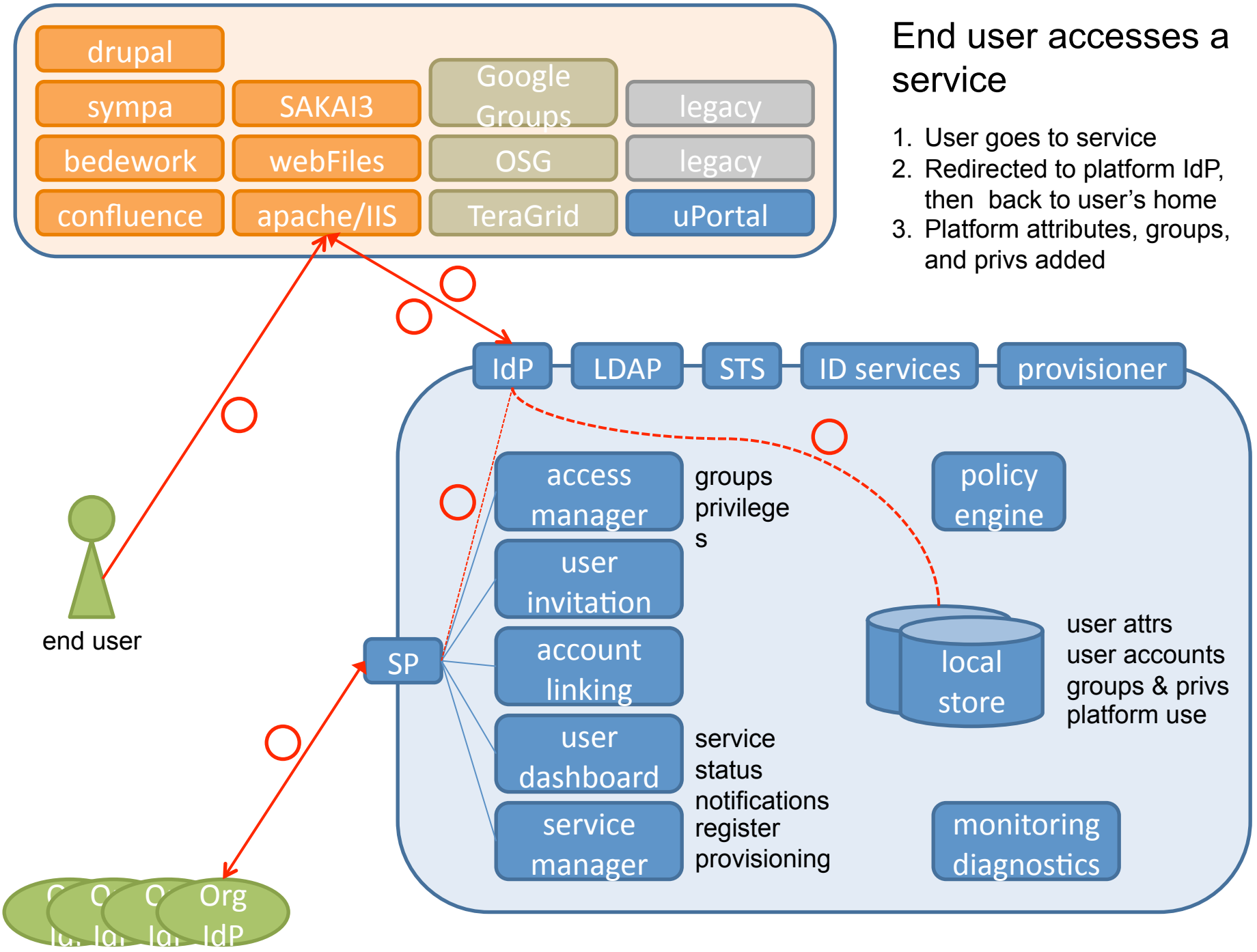

drupal
sympa
SAKAI3
Google Groups
legacy
bedework
webFiles
OSG
legacy
confluence
apache/IIS
TeraGrid
uPortal
Collabmin adds a new CO to the platform
1. Create group, assign Admin to power user
2. Allocate service resources
IdP
LDAP
STS
ID services
provisioner
access manager
groups
privileges
policy engine
user invitation
collabmin
SP
account linking
local store
user attrs
user accounts
groups & privs
platform use
user dashboard
service status
notifications
service manager
register
provisioning
monitoring
diagnostics
Org IdP

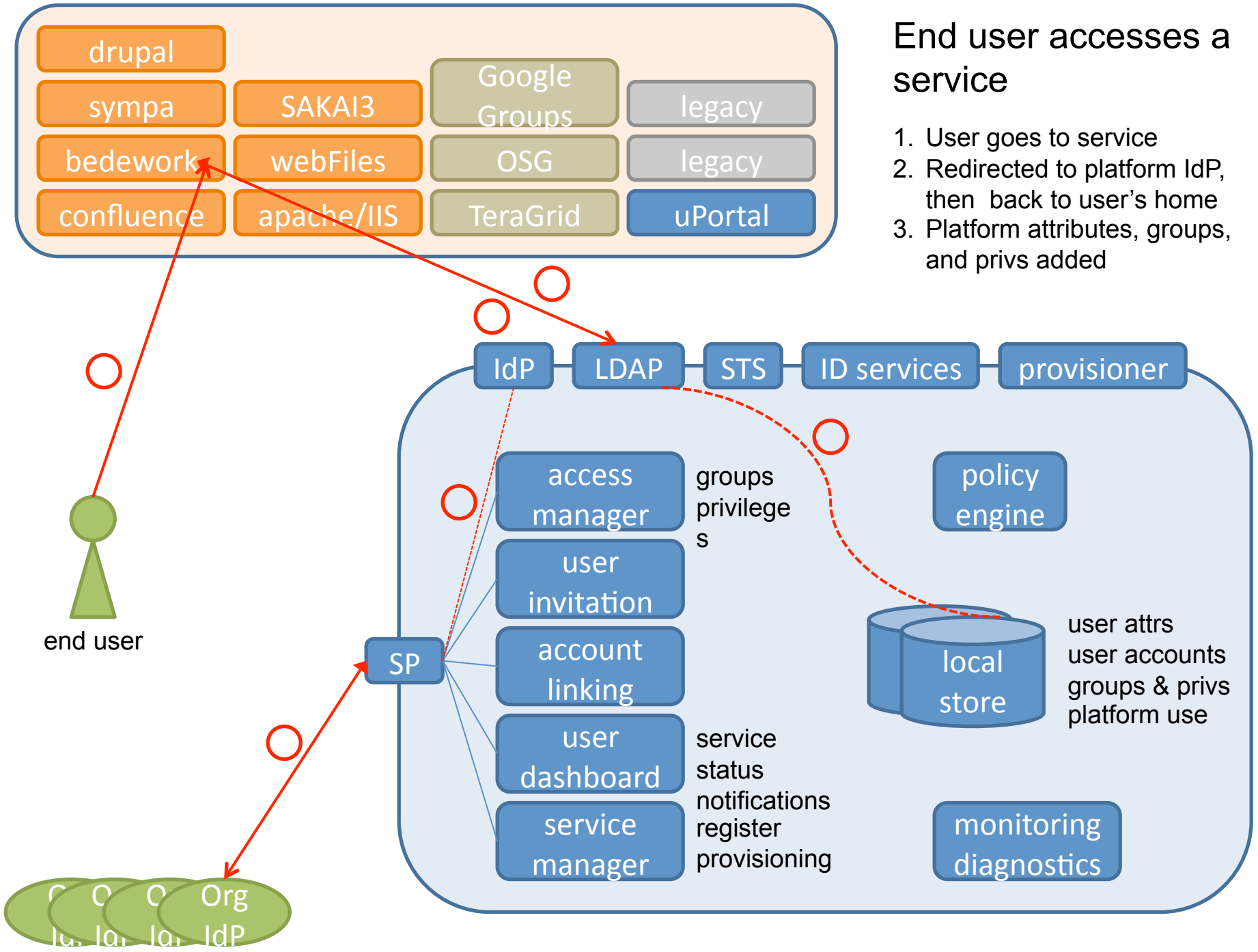


Power user invites a collaborator and gives them privileges

1. Invite user
2. Add user to CO group
3. User receives invitation token, presents it to invitation service to register with the platform




drupal
sympa
SAKAI3
Google Groups
legacy
bedework
webFiles
OSG
legacy
confluence
apache/IIS
TeraGrid
uPortal
End user accesses a service
1. User goes to service
2. Redirected to platform IdP, then back to user's home
3. Platform attributes, groups, and privs added
IdP
LDAP
STS
ID services
provisioner
access manager
groups
privileges
policy engine
user invitation
account linking
SP
local store
user attrs
user accounts
groups & privs
platform use
user dashboard
service status
notifications
service manager
register
provisioning
monitoring
diagnostics
end user
Org IdP


drupal
sympa
SAKAI3
Google Groups
legacy
bedework
webFiles
OSG
legacy
confluence
apache/IIS
TeraGrid
uPortal
End user accesses a service
1. User goes to service
2. Redirected to platform IdP, then back to user's home
3. Platform attributes, groups, and privs added
IdP
LDAP
STS
ID services
provisioner
access manager
groups
privileges
policy engine
user invitation
account linking
local store
user attrs
user accounts
groups & privs
platform use
SP
user dashboard
service status
notifications
service manager
register
provisioning
monitoring
diagnostics
end user
Org IdP

App developer framework

- Two types
  - Stand-alone app
  - Apps written in an application development environment, e.g. .NET or Spring or…
- Make clear that app data stays in app, not in comanage
- Presents a set of services – which ones

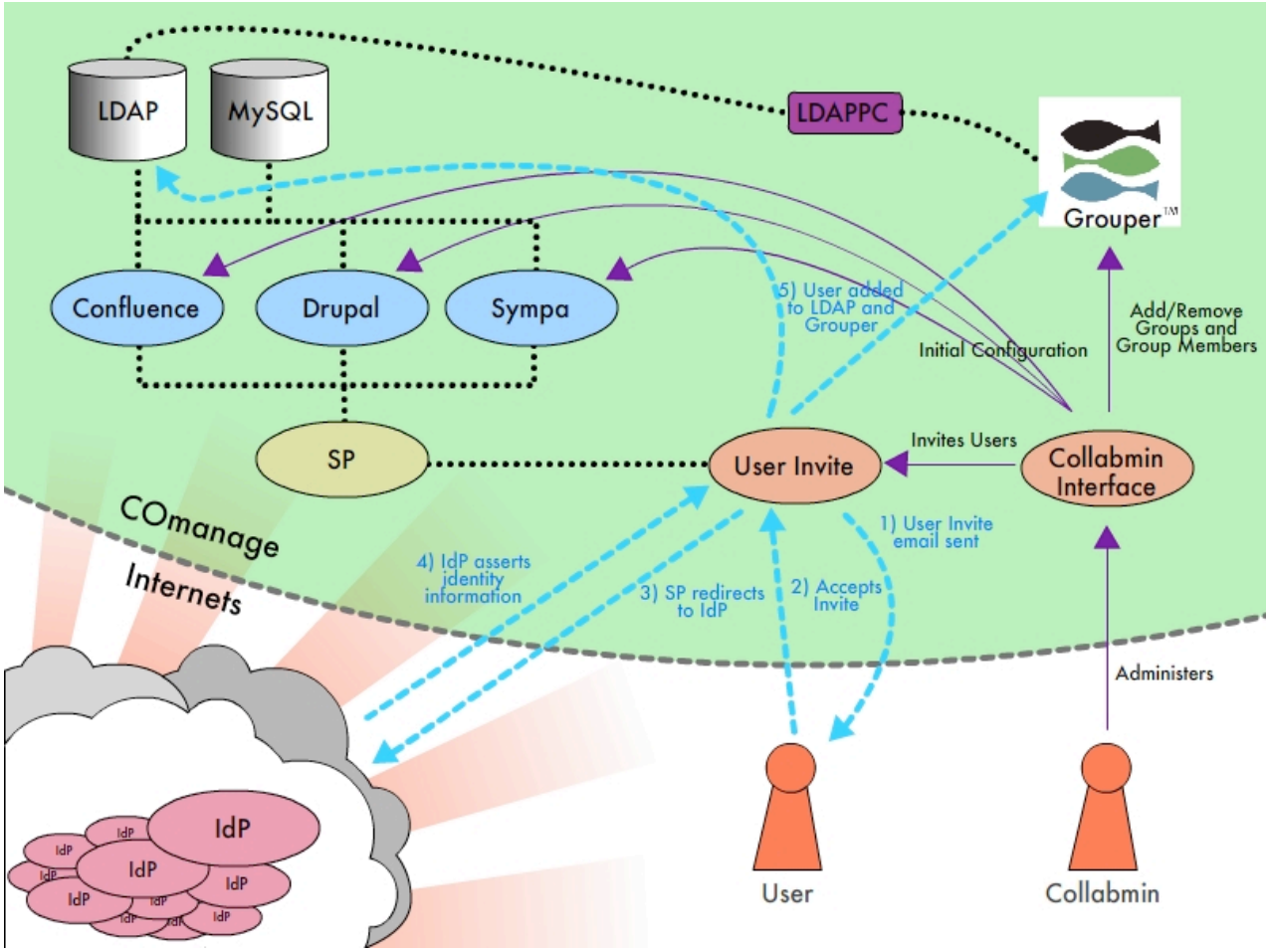
App developer framework

- Services provided are:
  - Authn
  - Authz (Y/N/?)
  - Attributes for app needs
  - Provisioning (?)
  - Some kind of monitoring
- Services explicitly not provided are:

How do apps get info

- Push into legacy apps
- Domesticated apps ask for it
- Domesticated apps need to speak LDAP or SAML or generic STS

Flows


LDAP
MySQL
LDAPPC
Grouper™
Confluence
Drupal
Sympa
5) User added to LDAP and Grouper
Initial Configuration
Add/Remove Groups and Group Members
SP
User Invite
Invites Users
Collabmin Interface
COmanage
Internets
4) IdP asserts identity information
3) SP redirects to IdP
2) Accepts Invite
1) User Invite email sent
Administers
IdP
User
Collabmin

Refactoring COmanage

- Right word for the concept?
  - Unbundling, debinding, distributing
- What are likely refactorings?
- What connections need to be in place among refactored pieces

Parked issues

- Discussion of how to share the work of domesticating apps
- Cutover issues for existing VO's, and type of collabs to target for appliance, etc
- Domesticated Zimbra - a lot of us are interested in it and claim to have connections with the company
- How might the appliance and an RSS feed offer a "collaboration stream”
- Maintaining a base level appliance
- Setting a new time for the COmanage dev calls
- Assess the viability of the existing appliance code base

More parked issues

- VOMS comparison/integration
- Licensing issues
- Application check-in services
- Developing use cases
- Is the proper technical phrasing “claims-aware”, “STS aware”, externalized or something else
-