LMS to myPortfolio

Interface Design

March 2011

Table of Contents

1. Executive Summary

1.1. Purpose of this document

The purpose of this document is to develop a design proposal for provisioning student, teacher, and parent/caregiver accounts in Mahara (myportfolio.school.nz) within the New Zealand compulsory sector education environment.

1.2. Identification

Modified by	*Date*	*Version*	*Reviewer/ Distribution*	*Notes*
Piers Harding	25/03/11	1	Paul Seiler MOE Architects Catalyst Dataview	First Draft
Piers Harding	30/03/11	2	MLE Team, Catalyst, Dataview	2nd Draft
Piers Harding	31/03/11	3	MLE Team, Catalyst, Dataview	3rd Draft – added section 6 – task list

1.3. Scope

The scope of this document is limited to the proposal and design of three mechanisms for the provisioning of accounts:

- Bulk account creation/update facility based on CSV file upload in the Mahara administration console.
- Automatic account provisioning, and account linking via the SAML based authentication model.
- A uni-directional Web Services interface between LMS (or any compliant system) and Mahara myPortfolio, for the provision of user accounts including Create, Update, and Delete functions.

1.4. Relationships to other documents

This document makes reference to :
SMS Identity Data Exchange Schema (Ministry/EdTech)
LMS Interoperability Solution Discussion (Ministry/Catalyst)
IMS Global General Web Services http://www.imsglobal.org/gws
IMS Global IMS Person Management http://www.imsglobal.org/es/esv1p0/imsperson_bindv1p0.html
JISC LEAP2A specifically personal, and organisation data http://wiki.leapspecs.org/2A/personal_data, http://wiki.leapspecs.org/2A/organizational_data
SMS-LMSv2 http://tech.groups.yahoo.com/group/data-sharing-framework/files/SMS-LMS%20Version%202%20%28Optional%29/
LMS myPortfolio Interoperability report http://wikieducator.org/LMS-MyPortfolio_Interoperability_Project

Moodle Web Services http://docs.moodle.org/en/Development:Web_services

1.5. Summary

The design proposed in this document has been developed as the result of an iterative process of researching, and experimenting with solutions that will fit the specific environment of the compulsory sector in New Zealand and LMS integration with the myportfolio.school.nz service.

It is understood that not all solutions proposed may be accepted, but the motivation is to describe what would produce an effective solution from the perspective of all actors in the network including Students, Teachers, Parent/Caregivers, School Administrators (or LMS managers), and the myPortfolio service administrators.

The focus is specifically on account management in myPortfolio, and providing fundamental structures in those accounts that make it possible for actors in both the LMS and ePortfolio to effectively map from one system to another.

In order to achieve this goal, the functionality required is not just account maintenance. Additionally, a set of structures are required that enable easy clustering of accounts from each users perspective, so that relationships between accounts (Parent/Caregiver ↔ Student, Teacher ↔ Student) can be easily established, and artefact access readily controlled.

These access rights are fundamentally the preserve of the Student, so it is imperative that the Student be given a way of identifying clusters of accounts that relate to them, and be in control of what access rights are allocated.

The research sources for this document include:

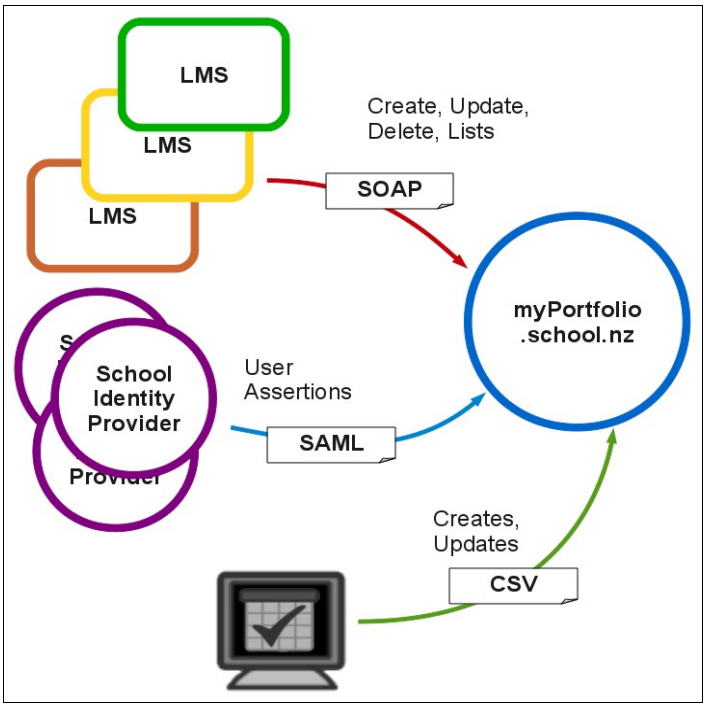
- IMS Global – GWS and ES specifications
- JISC - LEAP2A Enabling ePortfolio portability specification
- LMS myPortfolio Interoperability Project
- SMS LMSv2 Interoperability Project

2. Requirements

2.1. Overview

To be able to appeal to a broad base of LMS vendors, the aim is to adopt a standards based integration transport layers. However, there are multiple modes of integration, these being:

- automated remote provisioning, triggered by the LMS
- provisioning at time of user first contact via a school Identity service
- manual bulk account administration



2.1.1 LMS → ePortfolio Integration

The broadest supported set of standards likely to be adopted by the LMS vendors are the W3C Web Services standards centred on the SOAP messaging protocols. These standards are well defined and supported in any of the development environments associated with the core LMS vendors of New Zealand, and are likely to be the most common preference for a real-time back-office style integration strategy (there was no contradictory evidence out of the consultation process of the LMS Interoperability Design Workshop attended by LMS vendors in 2009).

The LMS to ePortfolio interface is uni-directional. All actions are to be triggered from the LMS, and access control over the API and actions available to each institution is controlled from within the ePortfolio.

2.1.2 User controlled provisioning

The Ministry of Education IAM initiative has adopted SAML as the authentication mechanism of choice. The SAML data model employed by the Ministry (as described by the IDE) carries sufficient information to enable the user to self provision on the fly, and to have the opportunity to amend their myPortfolio SSO institution relationship as they move between schools.

2.1.3 Manual account administration

The current bulk account management tools in Mahara, only extend to account creation based on CSV file upload. This is a useful and generic model for bulk account management, but could benefit from incorporating account updates, and exports via CSV upload/download, and single action bulk user updates via an enhanced user search facility. These enhancements are based on the experiences of the Moodle user administration tools which are enormously effective in this regard.

2.2. Account Types

There are three broad account types:

- Student
- Teacher
- Parent/Caregiver

Student and Teacher fall reasonably within the bounds of a standard Mahara account type, being able to take advantage of all standard functionality around an ePortfolio.
However, Parent/Caregiver only requires a limited subset of functionality, of a fundamentally 'observer' only nature (at least at this stage of development – it is conceivable that P/CG could contribute feedback in the future). This has lead to ideas around how to manage light weight account types, or even eliminate the need for established accounts at all in Mahara.
To achieve this there are a number of factors that must be taken into account with this:

- A student must retain control of their content – this includes being able to grant and more importantly revoke access as and when required.
- The application of access controls within Mahara naturally revolves around the notion of an account holder throughout the system. This covers everything from identifying and keeping track of who to give access to, through to how to control how those access restrictions are applied to the recipient.
- While Mahara will be developing mechanisms via the API to delegate authority to grant access to others, ultimately it should retain the ability and tools to manage the granting and revoking of access to all players from the API users, down to the individual user accessing content. This ability to control and audit access, would be severely compromised by anything other than having some form of authenticated account tracked from within Mahara.
- Zero footprint accounts require considerable integration with a 3rd party authentication system to make a reality, which will put constraints on which schools can participate.

2.2.1 Zero footprint accounts in more detail

In order to make zero footprint accounts a reality, then there are a number of operating conditions that must be met:

- The access control is enforced at every runtime (initial page request) as the the target system has no memory of the incoming entity requiring access.
- Mahara will have to trust (and have a trust relationship) with 3rd party LMS systems in order to establish the identity of the incoming entity.
- This access is inherently tied into the authentication process of both systems.
- The inter-system trust relationship is not specific to an individual entity, but a system as a whole. In this situation, Mahara is unable to make any value judgement around individuals, but relies solely on the 3rd party to enforce compliance with fair (or safe) use policy.
- The assertions made by the partner system need to be complex in order to describe the relationships between Students and Parent/Caregivers

Within the MLE framework, the mechanism that has the capability to implement this is SAML 2.0 based Web SSO. However, this – at least in the short to medium term – will present a barrier to entry for schools, who have not managed to go down this path yet (or may never be able to).

2.2.2 Recommendation

Given these essentially, structural issues for Mahara, the proposed design is to develop a new kind of account in Mahara that has observer/read only qualities. These accounts for Parent/Caregivers will be lightweight, and as such would have no quota or capabilities to create artefacts, or participate in forums. With these restrictions in place, this new account type should not produce significant overhead on the myPortfolio system. It also enables the effective decoupling of implementing support for Parent/Caregivers from external authentication models such as zero footprint accounts, while not eliminating this from future development plans, when the critical mass is established in the National MLE ecosystem.

2.3. Account Management Processes

Required account management processes by method are:

Web Services	Account Create, Update, Delete, list, revoke institution membership
SAML	Account Create, Update, Reassign Identity Provider
Bulk management tools	Create, Update, list, Reassign institution, and Identity Provider

2.3.1 Web Service Method

It is key that the scope of the actions that an LMS to myPortfolio pairing can perform are restricted to the Institution they are configured for.

This means that Web Services tied to one institution can create, update, and delete users in that institution but not others.
This creates issues around the notion of assigning users to institutions. One institution Web Service should not be able to switch membership of another institutions accounts. The danger here is that you could end up with inter-institution Web Service wars where each are switching accounts back and forth, leaving an account assigned incorrectly with respect to the appropriate authentication mechanism as well as the associated institution.

Switching accounts between institutions requires careful control because of the concern over who has the right to perform the operation. The best placed person is the account owner, and the current 'leave institution'/'accept institution membership request' process in Mahara best reflects this.

The Web Services (acting on behalf of the institution owner) should have the capability to revoke institution membership.

2.4. Account Relationship Processes

Two relationship management facilities:

- Groups used for class, and student group relationship mapping.
- Favourites used for Parent/Caregiver mapping.

2.4.1 Groups

The existing Mahara group membership facility has all the necessary basic principals for maintaining associations analogous to class, and interest groups in a schooling situation. These groups will only be maintainable via the API, by removing all group administrators.
Groups will need to be extended to provide additional flags so that they can be tracked to the institution that they belong to, and enable future updates to be channelled to the correct one, and enable them to be distinguished for searching and display purposes.

2.4.2 Favourites

In keeping with the philosophy described in the LMS myPortfolio Interoperability Project, a new feature will be introduced that enables the creation of 'Favourites'. These are similar to groups, with the sole purpose of maintaining collections of accounts known to the account holder, to make it easy for them to assign access rights to artefacts. Favourites do not require friendship to be mutual, and are entirely specific, and private to an individual user. Favourites will be controlled solely through the API, so there will be no manual management interface.

3. Web Services Design

In order to appeal to the widest range of 3rd party integrators, the technology choice for the transport layer must be universally supported. While not necessarily the most elegant, or pragmatic solution, the W3C Web Services specification for message exchange is arguably the most widely adopted, both in the commercial, and Open Source sectors of service providers.

W3C Web Services are new to Mahara. There is currently an XML-RPC implementation that is predominantly used for MNET communication with Moodle.
Modifying this to support a granularly controlled SOAP (http://www.w3.org/TR/soap12-part0/) implementation would require considerable effort, and has the added danger of breaking the existing solution that is widely used.

The preferred solution is to implement a new Web Services stack that is self contained within an artefact plugin, making it entirely optional.

Transport layer

Building a new Web Services stack is a medium to large undertaking. In order to reduce the lead time on this the proposal is to use the Moodle 2.x Web Services stack as a model, porting the code across where possible.

Aside from having a proven starting point for the design, there are a number of other benefits:

- The model is a framework that enables multiple responders eg: WS-SOAP, XML-RPC, REST, and AMF (Adobe action message format)
- Once implemented – developers need only create functions and register them
- It can be used as the basis for developing the notification delivery framework as ATOM/RSS

The framework will make the foundation for publishing any functionality within Mahara, opening the platform up to many other services – not just the LMS.

For all the benefits gained by starting with the Moodle Web Services stack, there are some key areas that will require adjusting:

- The configuration, and administration tools development process is fundamentally different so will have to be developed from scratch
- Moodle does not have the concept of Institution, which is will have to be built into the authentication model.
- Moodle Web Services do not currently support WS-Security extensions (Signatures, and Encryption). It is possible to secure the service by using SSL, but the security and integrity would be greatly enhanced by encryption and signatures.

Access Controls

Access control is to be implemented using the Moodle Web Services access token model. This is where a unique token is generated and attached to a specific Web Services-only user account. This account is allocated access rights to one or more institutions, and granted access to call specific functions. This will limit both the scope of actions that a 3rd Party application (LMS) can perform, and also which users (Institution specific) they can be performed against. The token forms part of the URL for the communication with the Web Service.

Example URL:

http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d

The administration tools for generation of the tokens, and assigning to privileged accounts, will also need to manage the allocation of these accounts to institution administration contacts.

Message Types

The description of the messages for account management are specific to Mahara, but where appropriate they have been linked to the key industry specifications of IMS-Global (Person/Organization), Leap2A and the Ministry of Education IDE and SMS-LMSv2.

The Moodle based Web Services framework will advertise WSDL (Web Service Description Language) descriptions for each service cluster relevant to the access token supplied.

See Appendix 1.

3.3.1 User Create/Update

Create and Update message types will be passed in arrays, allowing for multiple actions in a single request.

Visibility of user attribute values within Mahara is controlled at three levels:

- System wide config settings that determine whether a user profile value is mandatory or locked.
- Institution level config to set the locked status
- Account level to set the visibility of values within artefacts

Legend:

Types - M=Mandatory, MfC=Mandatory for Create, String=text object
IDE=Identity Data Extract

SMS-LMSv2 SmsStudentDemographics namespaces:

cbc=urn:nzl:govt:educating:draft:esl:schema:xsd:CommonBasicComponents-7
caccs=urn:nzl:govt:educating:draft:csl:schema:xsd:CommonAggregateComponents-6
cbccs=urn:nzl:govt:educating:draft:csl:schema:xsd:CommonBasicComponents-5

IMS-Global ES IMS Person Management Services do not readily map to Mahara – please refer to the linked data model for psm:

psm=http://www.imsglobal.org/es/esv1p0/imsperson_bindv1p0.html

Leap2A specifcation linked to persondata;

leap2a=http://wiki.leapspecs.org/2A/personal_data.

The relationship indicator is a guideline only. Implementers will still need to use their judgement to ensure the right source values are mapped.

Element	type	Description	relationship
username	Char(100) M	User name/Id – must be globally unique	IDE:mlepUsername, psm:identifier, leap2a:id[1]
password	Char(40) MfC	Password	
email	Char(255) MfC	Email address, must be valid email address format	IDE:mlepEmail, cbc:EmailEAddress, psm:Email, leap2a:email

[1] The LEAP2A id attribute type is relatively free form and can continue a multitude of id's, and is therefore designed to capture things like skype, AIM, ICQ, etc Ids

Element	type	Description	relationship
firstname	String MfC	First name	IDE:mlepFirstName, cbc:FirstName, psm:Name, leap2a:legal_given_name
lastname	String MfC	Last name	IDE:mlepLastName, cbc:LastName,psm:Name, leap2a:legal_family_name
institution	Char(255) MfC	Institution shortname identifier	Comes from the Mahara configuration for the target institution (blank for update means leave institution)
remoteuser	Char(255)	Remote User Id – typically used by SAML authentication	IDE:mlepGlobalUserId, or eduPersonPrincipalName
studentid	String	Student Id	IDE:mlepSmsPersonId, caccs:SMSStudentNumberID
preferredname	String	Preferred name	caccs:PersonNameGeneric where cbc:IsPreferredNameIndicator=true, leap2a:preferred_family_name+preferred_given_name
introduction	String	Introductory text	
officialwebsite	String	Official website	
personalwebsite	String	Personal website	cbc:WebsiteURLEAddress, leap2a:website
blogaddress	String	Blog URL	
address	String	Street address	caccs:PersonNZAddressUnstructuredGeneric where cbccs:AddressRoleTypeCode=Postal
town	String	Town	
city	String	City	
country	Char(2)	Lowercase ISO country code	For purposes of myPortfolio default to 'nz'
homenumber	String	Home phone number	caccs:PersonLandlineNumber, leap2a:homephone
businessnumber	String	Business phone number	caccs:PersonPhoneNumberGeneric where cbccs:PhoneRoleTypeCode=Work, leap2a:workphone
mobilenumber	String	Mobile phone number	caccs:PersonMobilePhoneNumber, leap2a:mobilephone
faxnumber	String	Fax number	caccs:PersonFaxNumber, leap2a:fax
icqnumber	String	ICQ chat Id	
msnnumber	String	MSN Chat Id	
aimscreenname	String	AIM screen name	
yahoochat	String	Yahoo Chat Id	
skypeusername	String	Skype user name	

Element	type	Description	relationship
jabberusername	String	XMPP (Jabber) user name	
occupation	String	Occupation	
industry	String	Industry	
maildisabled	Int 0 or 1	Flag for disabling email receipt	

3.3.2 User Create/Update Response

When a successful Create or Update request has completed, a message will be returned with an array of the following entries.

Element	type	Description
Userid	Int	Internal Mahara User Id
username	Char(100) M	User name/Id – must be globally unique

On any error, a SOAP fault will be issued with a relevant error message eg: invalid value, institution membership quota exceeded etc.

3.3.3 User Delete

Delete message types will be passed in arrays, allowing for multiple actions in a single request. There is only one element required, and that is the Mahara username.

Element	type	Description	relationship
username	Char(100) M	User name/Id – must be globally unique	IDE:mlepUsername, psm:identifier

3.3.4 User Delete Response

When a successful Create or Update request has completed, an empty SOAP message will be returned, with the usual HTTP 200 code.

On any error, a SOAP fault will be issued with a relevant error message eg: invalid username etc.

Example XML Messages can be found in Appendix 2.

4. SAML Integration Design

Existing SAML 2.0 support

The existing SAML (http://saml.xml.org/saml-technical-overview) authentication plugin in Mahara is not well enough designed to handle a transient user population. As students move between schools, it should be possible for them to manage their own account transition by being able to leave one institution, and request membership to another. Whilst this part of the process is possible at the institution membership level, it is not currently possible for a student to transition their SAML identity provider.

Additionally, it should also be possible:

- for manually created accounts, or existing accounts to be hooked up to an identity provider purely by logging in and linking accounts.
- New accounts created on the fly by initiating the login flow via an identity provider (first flow).

To achieve this, a number of new features are required that should only appear if the auth/saml plugin is active:

- Disconnect existing Identity Provider link
- Establish Identity Provider pairing
- First flow account creation
- Redirect users back to the deep URI that was originally requested (eg: forum post)

4.1.1 Disconnect existing Identity Provider link

Added to the user profile page, there should be two buttons. 'Disconnect Login Service', and 'Connect to Login Service'. These buttons should alternately appear depending on the current connection status. 'Disconnect Login Service' will remove the allocated remoteuser attribute (delete entry from auth_remote_user).

4.1.2 Disconnect existing Identity Provider link

'Connect to Login Service' will trigger the start of a SAML authentication flow for login-login-link.

Login-login-link is the process similar to establishing an OpenID pairing where a user logs into a service manually (username + password), and then logs into an authorised Identity Provider as well. The Identity provider declares a globally unique Id for the user that can be used to establish a permanent link to the new Identity Service used.
In this case, it will be used to establish a new remoteuser attribute tied to a specific institution SAML authentication service.

4.1.3 First flow account creation

If an account does not exist for a user that attempts to login via an institution's Identity Provider, then there should be sufficient information provided in the SAML assertions for an account to be created automatically.
Currently, when a user attempts to login via the SAML authentication plugin, if they login successfully at the IdP, but the attempt to match to a local Mahara account fails, the login process fails with a 'User not found' error. This will be amended to allow the user to either reconnect to an existing account (picking up that flow from 'Connect to Login Service' above), or opt to create a new account. The new account will complete the first part of the registration process automatically, as well as establishing the Identity Provider link, but there will still be a need for the user to complete the email portion of the self-registration process (see: http://myportfolio.school.nz/register.php).

Attributes mapped on user creation from the IAM IDE specification

Element	type	Description	Identity Data Extract attribute
username	Char(100) M	User name/Id – must be globally unique	mlepUsername
email	Char(255) MfC	Email address, must be valid email address format	mlepEmail,
firstname	String MfC	First name	mlepFirstName
lastname	String MfC	Last name	mlepLastName
institution	Char(255) MfC	Institution shortname identifier	Comes from the Mahara configuration for the target institution
remoteuser	Char(255)	Remote User Id – typically used by SAML authentication	mlepGlobalUserId, or eduPersonPrincipalName
studentid	String	Student Id	mlepSmsPersonId
country	Char(2)	Lowercase ISO country code	For purposes of myPortfolio default to 'nz'

4.1.4 Important points

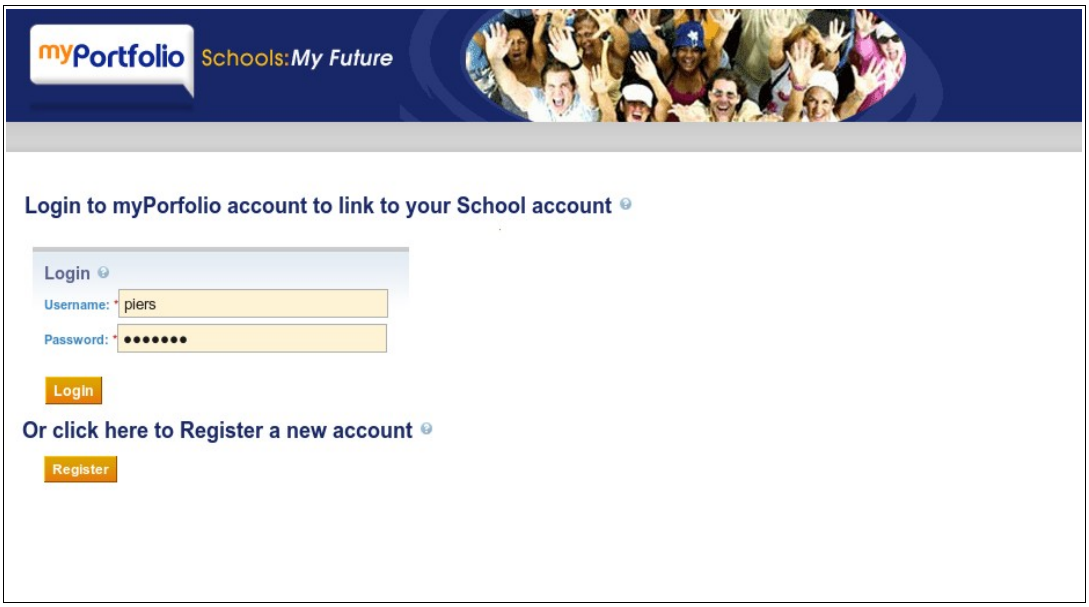
There must be a globally unique identifier to connect the user to the Identity Provider. Without this, it would be possible to have cross institution remoteuser name clashes. This should be the mlepGlobalUserId equivalent to the eduersonPrincipalName.

The users authentication method must be set to 'internal' so that it is possible for them to login manually, and via the Identity Provider. This also means that users must remember what their manual login password or be able to recover it, or they will not be able to manage their own disconnection/reconnection between institution login services.

4.1.5 Screen flow

The following are a set of screen mockups that illustrate the kind of screen flow that a user would encounter when signing in with SSO for the first time either with no myPortfolio account, or using a different School IdP:

Step 1. User has signed in with School IdP, and is returned to myPortfolio. Here the choice is to either sign in manually with an existing userid and password (branches to step 3), or to register a new account (continue to step 2).



Step 2. User Chooses to register a new account, whilst remaining logged in via the IdP

Step 3. With either a newly registered, or existing account, the user confirms linking the myPortfolio account with the SSO account from the School IdP.

5. Account Management Tools Design

Mahara has essentially only one mechanism for bulk account creation currently, and this is the CSV upload facility.

Individual accounts can be uploaded using the Leap2A format, but this is still managed on an account by account basis.

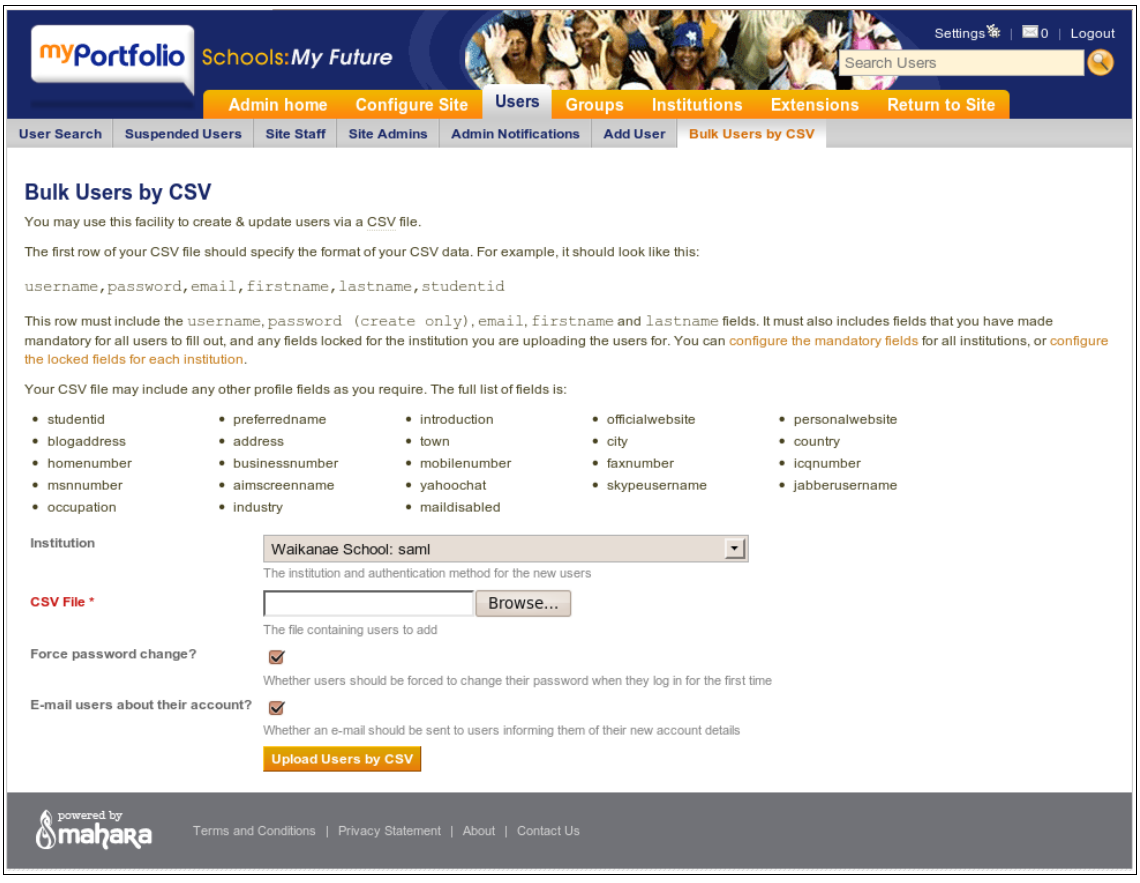
There are currently 134 institutions in myPortfolio with more than 5 accounts, with a grand total of 24,500 accounts. With an average of 180 accounts per school, this is likely to be a significant administration overhead, that will only continue to grow.

The following is a set of features that would greatly improve the manual management situation for individual institution Administrators:

- Bulk update of basic account details – synchronising with the school directory and/or LMS.
- Bulk update for leaving institution – being able to remove students from institution at year end
- Bulk update of Identity Provider associations for existing accounts and roll ons of students moving between schools (as an alternative to the SAML authentication plugin process described above).
- Bulk export of user list to help reconcile back to school SMS/directory, and provide the basis for input to bulk updates.

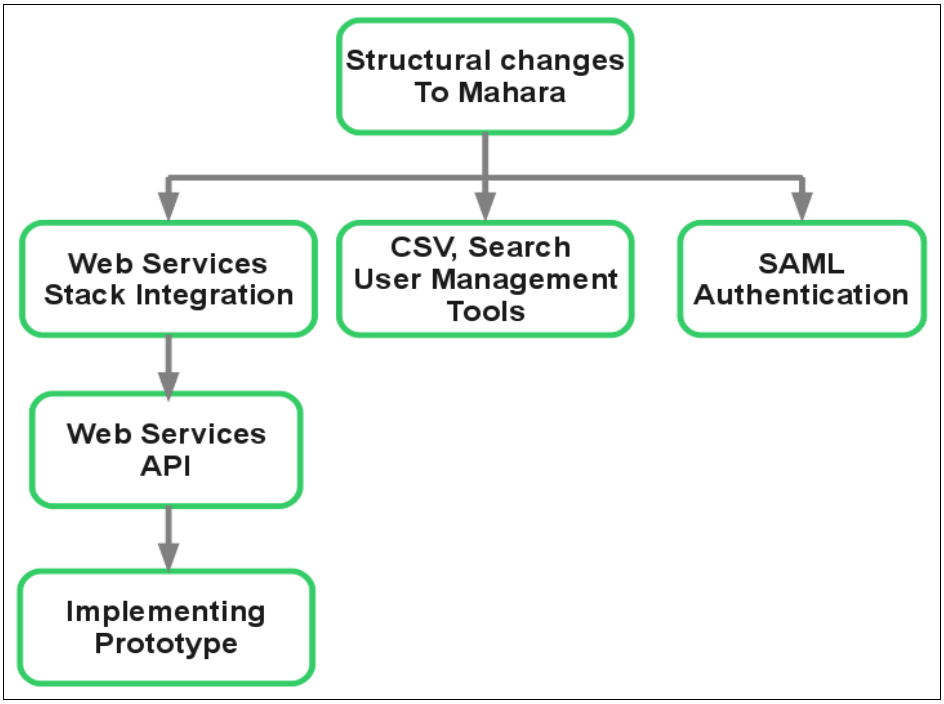
The necessary CSV file format is identical to element list described in section 3.3 Message types – User Create/Update (with the exception of password for export).

Mockup of amended interface in myPortfolio



6. Task List

The Phase one implementation task list is broken down into 6 major groupings. The dependencies and potential for breaking the tasks into work streams are outlined in the following diagram.



The main work flow is dependent on the Structural changes to Mahara, but after this, the remaining tasks can be broken into 3 separate streams, that can be worked on autonomously.

Structural changes to Mahara

- Parent/Caregiver read only accounts
- Favourites
  - add favourites data model.
  - add favourites as a selector for sharing like 'Share with other users and groups'
- API controlled Groups - make groups read only on this basis
- User data model changes for API support, and SAML support (self switching accounts)

CSV and Search User Management Tool changes

- Include specification of remote user to add
- Enable filtering of authentication types by either institution or authentication type (this is becoming confusing with the proliferation of both)
- Enable update of users via CSV upload - must match on username
  - check into access controls over different institution administrators updating different accounts
  - check who can add a user to an institution.
  - Check who can change them - limited fields
- Use update to do bulk institution assignments, and bulk authentication updates (set institution authentication, and remote user)
- Bulk suspend, bulk delete (work off the user search)

- Bulk export in default CSV format, as an aid to further bulk updates (this will hang off search)

SAML authentication improvements

- Standardise on EPPN - remove separate organisation, and username field requirements in configuration
- Dual login – SimpleSAMLphp, and Mahara session integration for login-login-link (lll)
- Link or register page
- On login for link, take to link confirmation page
- Preserve deep URI (contact URI)
- Pre-populate user registration with IdP values on registration page. Once registered, take to link confirmation process (as above).

Web Services stack integration - based on Moodle 2.0 Web Services

- Port over the Web Services data model - integrate for install/upgrade
- Port over code base to sit in artefact framework
- Create framework for registering services (e.g. db/service.php system)
- Activate SOAP, XML-RPC, and REST frameworks (disable AMF)
- Modify data model to support Mahara users, and the notion of Institutions - access for authorise tokens must be tied to an institution (this is precaution)
- Port Web Service test client
- Implement WS-Security encryption, and signatures
- Test suite - cover basic transport types execution, and access token validity (user/institution)
- Redevelop administration controls
  - enable/disable Web Services
  - create/delete service - assign user, and generate access token
  - allocate/deallocate registered functions to service

This should have an additional transport layer implemented for RSS/ATOM - this will be necessary for event notification.

Web Services API

All Web Services should be institution specific - access tokens are restricted to administrator user/institution .

- User create/update/delete - handles institution authentication allocation
- Grant/revoke institution membership
- Create/update/delete API controlled groups - update includes user membership grant/revoke
- Add/remove API controlled favourites

Implementing Prototype

- Upgrade myPortfolio
  - notify and educate existing school user base of the new administration tools, and SSO capabilities
  - SSL enable myportfolio.school.nz
  - upgrade myPortfolio
  - migrate existing SAML SSO enabled schools configuration to new implementation
- Build example client programs that demonstrate implementation pattern for Web Services API (SOAP)
- Develop integration guide for implementers
- Provide test environment+service to Dataview
- General testing assistance
- Activation of live service+monitoring
- Ongoing support

7. Appendices

WSDL Description example

An example of what an automatically generated WSDL file. The WSDL is accessed by calling the Web Service URL with the access token, and the additional WSDL request parameter, eg:

http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d&wsdl=1

```
<?xml version="1.0"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap-enc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
name="webservices_virtual_class_000000"
targetNamespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d">

 <types>
  <xsd:schema targetNamespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d" />
 </types>
 <portType name="webservices_virtual_class_000000Port">
  <operation name="mahara_user_update_users">
   <input message="tns:mahara_user_update_usersIn" />
  </operation>
  <operation name="mahara_user_create_users">
   <input message="tns:mahara_user_create_usersIn" />
   <output message="tns:mahara_user_create_usersOut" />
  </operation>
  <operation name="mahara_user_get_users_by_id">
   <input message="tns:mahara_user_get_users_by_idIn" />
   <output message="tns:mahara_user_get_users_by_idOut" />
  </operation>
  <operation name="mahara_user_delete_users">
   <input message="tns:mahara_user_delete_usersIn" />
  </operation>
 </portType>
 <binding name="webservices_virtual_class_000000Binding"
 type="tns:webservices_virtual_class_000000Port">
  <soap:binding style="rpc"
  transport="http://schemas.xmlsoap.org/soap/http" />
  <operation name="mahara_user_update_users">
   <soap:operation soapAction="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d#mahara_user_update_users" />
   <input>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </input>
   <output>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </output>
```

```
  </operation>
  <operation name="mahara_user_create_users">
   <soap:operation soapAction="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d#mahara_user_create_users" />
   <input>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </input>
   <output>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </output>
  </operation>
  <operation name="mahara_user_get_users_by_id">
   <soap:operation soapAction="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d#mahara_user_get_users_by_id" />
   <input>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </input>
   <output>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </output>
  </operation>
  <operation name="mahara_user_delete_users">
   <soap:operation soapAction="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d#mahara_user_delete_users" />
   <input>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </input>
   <output>
    <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?wstoken=484dfea8715ed427b4d95796c3013f7d" />
   </output>
  </operation>
 </binding>
 <service name="webservices_virtual_class_000000Service">
  <port name="webservices_virtual_class_000000Port"
  binding="tns:webservices_virtual_class_000000Binding">
   <soap:address location="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d" />
  </port>
 </service>
 <message name="mahara_user_update_usersIn">
  <part name="users" type="soap-enc:Array" />
 </message>
 <message name="mahara_user_create_usersIn">
  <part name="users" type="soap-enc:Array" />
 </message>
 <message name="mahara_user_create_usersOut">
  <part name="return" type="soap-enc:Array" />
 </message>
 <message name="mahara_user_get_users_by_idIn">
  <part name="userids" type="soap-enc:Array" />
```

```
 </message>
 <message name="mahara_user_get_users_by_idOut">
  <part name="return" type="soap-enc:Array" />
 </message>
 <message name="mahara_user_delete_usersIn">
  <part name="userids" type="soap-enc:Array" />
 </message>
</definitions>
```

SOAP Message examples

Example request/response messages are without signatures, and encryption.

Example Create User request:

```
<?xml version="1.0" encoding="utf-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns1="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d"
xmlns:ns2="http://xml.apache.org/xml-soap"
xmlns:enc="http://www.w3.org/2003/05/soap-encoding"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <env:Body>
  <ns1:mahara_user_create_users env:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
   <users enc:itemType="ns2:Map" enc:arraySize="1"
    xsi:type="enc:Array">
     <item xsi:type="ns2:Map">
      <item>
       <key xsi:type="xsd:string">username</key>
       <value xsi:type="xsd:string">bean2</value>
      </item>
      <item>
       <key xsi:type="xsd:string">password</key>
       <value xsi:type="xsd:string">bean</value>
      </item>
      <item>
       <key xsi:type="xsd:string">firstname</key>
       <value xsi:type="xsd:string">bean</value>
      </item>
      <item>
       <key xsi:type="xsd:string">lastname</key>
       <value xsi:type="xsd:string">bean</value>
      </item>
      <item>
       <key xsi:type="xsd:string">email</key>
       <value xsi:type="xsd:string">bean2@local.net</value>
      </item>
      <item>
       <key xsi:type="xsd:string">institution</key>
       <value xsi:type="xsd:string">wahoodle</value>
      </item>
      <item>
       <key xsi:type="xsd:string">auth</key>
       <value xsi:type="xsd:string">internal</value>
      </item>
      <item>
       <key xsi:type="xsd:string">mailformat</key>
       <value xsi:type="xsd:string">0</value>
      </item>
```

```
        <item>
          <key xsi:type="xsd:string">description</key>
          <value xsi:type="xsd:string">The Bean</value>
        </item>
        <item>
          <key xsi:type="xsd:string">city</key>
          <value xsi:type="xsd:string">Beanville</value>
        </item>
        <item>
          <key xsi:type="xsd:string">country</key>
          <value xsi:type="xsd:string">nz</value>
        </item>
      </item>
    </users>
  </ns1:mahara_user_create_users>
 </env:Body>
</env:Envelope>
```

General SOAP fault document:

```
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
   <SOAP-ENV:Fault>
     <faultcode>Receiver</faultcode>
     <faultstring>debug/invalidparameter</faultstring>
   </SOAP-ENV:Fault>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Success response:

```
<?xml version="1.0" encoding="utf-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns1="http://mahara.local.net/maharadev/artefact/webservice/soap/server.php?
wstoken=484dfea8715ed427b4d95796c3013f7d"
xmlns:ns2="http://xml.apache.org/xml-soap"
xmlns:enc="http://www.w3.org/2003/05/soap-encoding"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <env:Body xmlns:rpc="http://www.w3.org/2003/05/soap-rpc">
   <ns1:mahara_user_create_usersResponse env:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
    <rpc:result>return</rpc:result>
     <return enc:itemType="ns2:Map" enc:arraySize="1"
     xsi:type="enc:Array">
       <item xsi:type="ns2:Map">
         <item>
           <key xsi:type="xsd:string">id</key>
           <value xsi:type="xsd:int">18</value>
         </item>
         <item>
           <key xsi:type="xsd:string">username</key>
           <value xsi:type="xsd:string">bean2</value>
         </item>
       </item>
     </return>
   </ns1:mahara_user_create_usersResponse>
 </env:Body>
</env:Envelope>
```