PassUr1 : gestion des utilisateurs passagers de l'Université de Rennes 1

Pascal Aubry
IFSIC – Université de Rennes 1
Version 0.3 – 10 mai 2004

Statut

Diffusé à la liste projet-ldap, en attente de retours.

Ce document est accessible sur le web :

Au format HTML : http://perso.univ-rennes1.fr/pascal.aubry/doc/passur1 ;

Au format PDF : http://perso.univ-rennes1.fr/pascal.aubry/doc/passur1/passur1.pdf.

Historique

Version 0.3 – 10 mai 2004 :

Précisions sur les mises à jour d'attributs dans l'annuaire LDAP à partir des caractéristiques des passagers dans la base de données PassUr1.

Version 0.2 – 6 mai 2004 :

Expression de la stratégie générale de la solution et des flux d'information.

Version 0.1 – 28 avril 2004 (P. Aubry – IFSIC – Université de Rennes 1)

Table des matières

1. Motivations et objectifs

L'annuaire LDAP dont il est question dans ce document est celui de l'Université de Rennes 1, tel qu'il sera à la rentrée 2004, c'est-à-dire compatible SupAnn.

Les données de l'annuaire LDAP de l'Université sont issues de :

Apogée pour les étudiants ;

Harpège pour les personnels ;

Saisies manuelles pour les comptes dits « comptes manuels ».

Il faut, pour bien saisir l'intérêt de ce document, se convaincre du rôle de chaque acteur :

Les bases de données Apogée et Harpège sont le cœur du système d'information de l'Université pour ce qui concerne les utilisateurs.

L'annuaire LDAP est l'outil permettant de contrôler l'accès des utilisateurs de l'Université aux ressources informatiques.

1.1. Pallier certaines insuffisances du S.I. actuel

Pour de multiples raisons, certains utilisateurs de ressources ne peuvent pas être entrés dans le système d'information (parce qu'ils ne sont ni étudiant ni personnel de l'Université, parce que leur place n'a pas été prévue dans les bases de données et/ou les applications actuelles, etc.).

Pour quand même donner accès à ces **utilisateurs passagers** aux ressources informatiques, il est donc nécessaire d'ajouter au système d'information une troisième source de données. C'est dans cette source de données que seront stockés les utilisateurs manquant dans le S.I., et à partir de laquelle seront produites de nouvelles données dans l'annuaire LDAP, correspondants aux utilisateurs manquants.

Cette source de données supplémentaire est appelée PassUr1, pour « Passagers de l'Université de Rennes 1 ».

1.2. Améliorer la sécurité et l'interfaçage de l'accès à l'annuaire LDAP

La gestion des utilisateurs banalisés est aujourd'hui faite par un accès direct à l'annuaire LDAP. Cela oblige :

A ouvrir des accès directs à l'annuaire à tous les administrateurs devant effectuer des opérations, ce qui abaisse la sécurité de l'annuaire ;

Effectuer des manipulations directes dans l'annuaire, ce qui est source d'erreur et difficilement traçable.

Pour améliorer la **sécurité**, la **fiabilité** et la **traçabilité**, l'annuaire doit être interfacé par une application.

Cette application porte le nom de la base de données qu'elle gère, PassUr1.

1.3. Déléguer la gestion de certains utilisateurs aux composantes

La gestion des utilisateurs banalisés, ainsi que la saisie et la gestion des utilisateurs banalisés, doit être déléguée à des personnels choisis des composantes, dans un double objectif :

Simplifier les tâches d'administration du CRI ;

Améliorer le service aux utilisateurs dans les composantes, par plus de réactivité.

La délégation sera obtenue par la mise à disposition de l'application PassUr1 dans les composantes.

2. Intégration dans le S.I actuel

2.1. Architecture

La figure 1 montre l'intégration de cette troisième source de données dans le S.I. actuel en matière d'architecture.

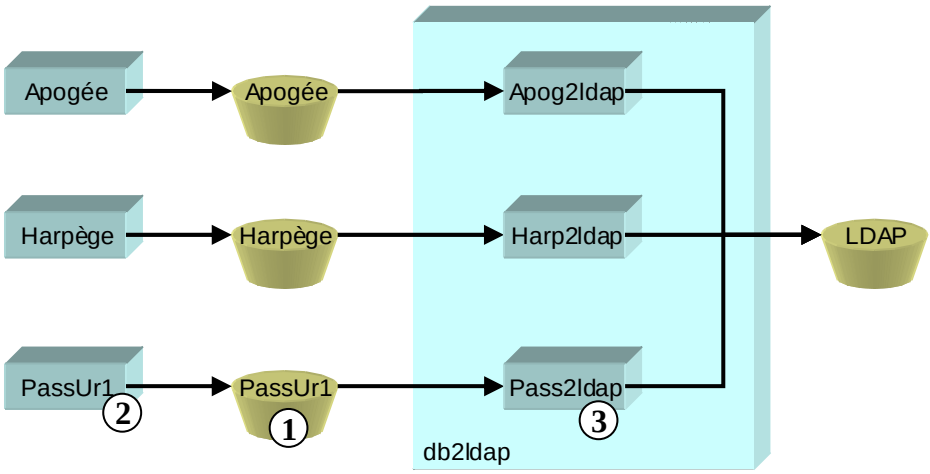

figure 1 Architecture globale de la solution

La mise en œuvre de la solution implique :

La mise en place d'une base de données (1) ;

L'écriture de l'application permettant de saisir les utilisateurs dans la base de données (2) ;

La modification de la passerelle (Harpège+Apogée) vers LDAP actuelle pour prendre en compte les utilisateurs passagers (3).

2.2. Stratégie générale

Cette partie montre la stratégie de la solution proposée, à travers les flux d'information.

2.2.1. Flux d'informations

La stratégie de gestion des utilisateurs passagers est directement héritée de celle des utilisateurs inscrits dans les bases Harpège et Apogée :

Les caractéristiques des utilisateurs, comme par exemple l'identité et les inscriptions administratives, sont créées et maintenues dans la base PassUr1, à l'aide de l'application PassUr1 ;

Elles sont créées et mises à jour dans l'annuaire LDAP par la passerelle Pass2ldap ;

Les autres informations, telles le mot de passe ou le shell des utilisateurs, sont mises à jour par d'autres applications (Sésame par exemple).

Notons que les informations concernant les passagers doivent être :

Suffisantes pour maintenir les informations de l'annuaire LDAP qui ne sont pas maintenues par d'autres applications, afin de supprimer les accès directs à LDAP pour la gestion des passagers ;

Nécessaires seulement pour éviter toute redondance de données.

2.2.2. Réduction de l'asynchronisme

La différence essentielle avec les schémas adoptés par Apogée et Harpège concerne le mode de propagation des informations vers l'annuaire LDAP. Les passerelles de ces bases vers l'annuaire LDAP sont exécutées

de manière asynchrone, quotidiennement. En effet, les applications Apogée et Harpège, développées par l'AMUE, sont fermées ; la solution mise en place pour propager les données dans l'annuaire LDAP est de :

Chercher les opérations effectuées dans les bases ;

Les répercuter dans l'annuaire.

C'est bien le coût de la recherche des ajouts/modifications qui limite la fréquence d'exécution des passerelles, alors que les opérations de mise à jour de l'annuaire en elles-mêmes sont très simples. Pour éviter ces recherches, le schéma suivant est adopté :

A chaque ajout ou modification d'une information dans la base PassUr1, l'événement est enregistré dans la base PassUr1, sous forme d'une notification ;

La passerelle Pass2ldap accède directement à la liste des notifications, effectue les opérations correspondantes dans l'annuaire, et note dans la base de données les notifications comme traitées. De cette manière, la passerelle peut être exécutée de manière très fréquente, puisque la plupart du temps, aucune opération ne sera à effectuer.

La prise en compte des changements est certes effectuée de manière asynchrone, mais la fréquence d'exécution peut certainement être considérablement augmentée : en gérant un verrouillage pour éviter les instances multiples de la passerelle, celle-ci peut sans aucun doute être lancée toutes les minutes par une tâche planifiée. L'asynchronisme serait alors quasi invisible.

3. Les Objets métiers

Pour chaque objet métier, on précise s'il est stocké en base de données ou défini (déclaré au format XML dans un fichier). On indique également pour chacune de ses caractéristiques :

Son type ;

Sa sémantique ;

Un exemple ;

Qui l'utilise et comment, si nécessaire ;

Qui le met à jour, et à quelle occasion, si nécessaire.

L'implémentation de l'application PassUr1 et de la passerelle db2ldap étant envisagée en Java, avec utilisation de JDO pour le mappage dans la base de données, les types des objets sont donnés en Java, types dont on pourra déduire l'équivalent JDBC (cf http://db.apache.org/ojb/jdbc-types.html).

3.1. Les composantes

Les composantes sont utilisées pour :

Les utilisateurs, qui peuvent être gestionnaires des passagers d'une ou plusieurs composantes ;

Les profils d'utilisateurs, qui ne sont utilisables que par les gestionnaires d'une composante.

Les composantes sont représentées par des instances de la classe CPassUr1Department.

3.1.1. Déclaration

Les données représentant les composantes étant très stables, la liste des composantes est stockée au format XML, dans un fichier de configuration, et non modifiable par l'application PassUr1.

La modification de la liste des composantes se fait donc par l'édition d'un fichier, au format XML, par les administrateurs de l'application. Cela évite ainsi de développer une interface de gestion des composantes.

Toutes les instances de la classe CPassUr1Department sont créées au démarrage des applications qui en ont besoin (PassUr1 et db2ldap).

Note : Il serait évidemment intéressant de pouvoir lier directement les données représentant les composantes au S.I. existant. Une possibilité pour maintenir au moins une cohérence sans lien direct serait de générer automatiquement et périodiquement les données à partir du S.I., de manière périodique.

3.1.2. Caractéristiques

- *Id (String)*

Id est un identifiant unique, égal au code identifiant la composante dans le S.I. de l'Université (exemple : 913 pour l'IFSIC).

- *Label (String)*

Label est une chaîne de caractères décrivant brièvement la composante (exemple : IFSIC).

- *Managers (Vector of String)*

Managers est un vecteur donnant les identifiants des gestionnaires de la composante.

3.2. Les gestionnaires

Du point de vue de la solution, quelques personnels de l'Université sont privilégiés, habilités à gérer les passagers d'une ou plusieurs composantes.

On les appelle les gestionnaires des composantes.

3.2.1. Déclaration

Les données représentant les gestionnaires sont, comme celles des composantes, très stables. Pour cette raison, et pour ne pas avoir besoin d'une interface de gestion (ajout/suppression) des gestionnaires des composantes, la définition des gestionnaires stockée au format XML, dans un fichier de configuration, et non modifiable par l'application PassUr1.

La modification de la liste des composantes se fait donc par l'édition d'un fichier, au format XML, par les administrateurs de l'application.

Dans une version ultérieure, les gestionnaires pourront être déduits de l'annuaire LDAP grâce à la remontée d'un attribut multi-valué par uPortal.

3.2.2. Caractéristiques

Les gestionnaires des composantes sont simplement repérés par leurs uids LDAP.

3.3. Les profils de passager

Certaines informations caractérisant les passagers sont nominatives, c'est-à-dire qu'elles ne concernent que l'individu caractérisé (par exemple l'identité).

En revanche, d'autres peuvent être partagées entre plusieurs individus, autorisant ainsi une gestion des passagers par groupes. L'ensemble des informations partagées entre plusieurs passagers est appelé profil de passager.

A l'inverse, un passager est lié à un profil, c'est-à-dire qu'il « hérite » de toutes les propriétés du profil auquel il est lié.

Notons que :

> Un passager peut changer de profil, mais un passager étudiant (resp. personnel) est toujours lié à un profil étudiant (resp. personnel) ;

Un profil est lié à une composante, c'est-à-dire qu'il n'est visible et utilisable que par les gestionnaires d'une composante.

3.3.1. Stockage

Les données représentant les profils sont assez volatiles. Elles sont mappées dans la table profile de la base de données PassUr1, et gérées à travers l'application PassUr1.

3.3.2. Caractéristiques

La mise à jour d'une des caractéristiques employeeType, departmentNumberList, ur1ComposanteList ou ur1InscriptionList crée à l'intention de la passerelle Pass2ldap des notifications de type USER_CHANGE, pour tous les utilisateurs liés au profil.

- *Id (long)*

Id est un identifiant unique, incrémenté automatiquement.

- *Label (String)*

L'identifiant alphabétique permettant de repérer facilement les profils entre eux.

Utilisée dans l'application PassUr1 pour ne pas afficher l'identifiant numérique.

Mise à jour par l'application PassUr1.

Exemples : 2004-ifsic-etud-msi2, 2004-ifsic-pers-ext

- *Department (String)*

L'identifiant de la composante à laquelle est lié le profil. Les valeurs admises sont configurées dans l'application PassUr1.

Utilisée dans l'application PassUr1 pour trouver les profil des composante.

Initialisée à la création du profil dans l'application PassUr1, cette caractéristique ne peut être ensuite modifiée.

Exemple : 913 pour l'IFSIC.

- *Student (boolean)*

Vrai si le profil correspond à des passagers étudiants, faux sinon.

Utilisée par la passerelle Pass2ldap, en particulier pour savoir comment générer l'uid dans l'annuaire LDAP et maintenir l'attribut ur1TypeEntree des passagers du profil.

Initialisée à la création du profil dans l'application PassUr1, cette caractéristique ne peut être ensuite modifiée.

- *EmployeeType (String)*

Une chaîne de caractères représentant le type des passagers du profil. Les valeurs admises sont configurées dans l'application PassUr1 (séparément pour les passagers étudiants et personnels).

Utilisée par la passerelle Pass2ldap pour maintenir l'attribut employeeType de l'annuaire LDAP.

Mise à jour par l'application PassUr1.

Exemple : IATOS

- *DepartmentNumberList (String)*

Une liste de chaînes de caractères représentant des entités de l'Université, séparées par des virgules.

Utilisée par la passerelle Pass2ldap pour maintenir l'attribut multi-valué departmentNumber de l'annuaire LDAP.

Mise à jour par l'application PassUr1.

Exemple : UR1,957,57SI

- *Ur1ComposanteList (String)*

Une liste de chaînes de caractères représentant des entités de l'Université, séparées par des virgules.

Utilisée par la passerelle Pass2ldap pour maintenir l'attribut multi-valué ur1composante de l'annuaire LDAP.

Mise à jour par l'application PassUr1.

Exemple : 922,957

- *Ur1InscriptionList (String)*

Une liste de chaînes de caractères représentant des inscriptions administratives d'étudiants à l'Université, séparées par des virgules.

Utilisée seulement pour les passagers étudiants, par la passerelle Pass2ldap, pour maintenir l'attribut multi-valué ur1inscription de l'annuaire LDAP.

Cette caractéristique n'est employée que pour les passagers étudiants.

Exemple : P:2003:913:S30031:3:E pour une inscription en DESS ISA en 2003.

- *Date (java.sql.Date)*

La date à laquelle les comptes de tous les passagers liés au profil seront automatiquement fermés. Note : les comptes peuvent être fermés par anticipation pour des utilisateurs particuliers en modifiant la caractéristique active des passagers.

Utilisée par la passerelle Pass2ldap, pour fermer les comptes des utilisateurs à échéance.

Mise à jour par l'application PassUr1. Vérifier avec Claude si une notification est nécessaire.

3.4. Les passagers

La plupart des informations concernant les passagers sont maintenues dans le profil auquel est lié le passager. Les informations restantes sont nominatives, c'est-à-dire maintenues au niveau des données des passagers.

3.4.1. Stockage

Les données représentant les profils sont assez volatiles. Elles sont mappées dans la table user de la base de données PassUr1, et gérées à travers l'application PassUr1.

La création d'un passager dans la base de données crée une notification de type USER_CREATE.

3.4.2. Caractéristiques

- *Id (long)*

Id est un identifiant unique, incrémenté automatiquement.

- *Uid (String)*

L'uid du passager dans l'annuaire LDAP.

Utilisée par la passerelle Pass2ldap pour faire l'association entre les entrées dans la base de données PassUr1 et dans l'annuaire LDAP.

Vide à la création du passager, renseignée par la passerelle Pass2ldap lors du choix de l'uid, ne peut plus être modifiée ensuite.

Exemple : paubry.

- *UsualName (String), GivenName (String) et PatronymicName (String)*

Le nom usuel, le prénom et le nom patronymique (de jeune fille) du passager.

Utilisées par la passerelle Pass2ldap pour maintenir les attributs suivants de l'annuaire LDAP :

usualName : cn, gecos, sn, et ur1nomusuel

givenName : cn, gecos, givenname, et ur1prenom

patronymicName : sn, cn et ur1nompatronymique

Mises à jour par l'application PassUr1, ce qui crée une notification de type USER_CHANGE.

Exemples : AUBRY, PASCAL, AUBRY

- *Active (boolean)*

Vrai si le compte est actif, faux sinon.

Utilisée par l'application PassUr1 pour maintenir l'état du compte d'un passager. Note : un compte peut de cette manière manière être marqué inactif même si la date de validité du profil n'est pas dépassée, ce qui permet de prendre en compte les abandons ponctuels d'étudiants.

Mise à jour par l'application PassUr1, ce qui crée à l'intention de la passerelle Pass2ldap un événement de type USER_ CLOSE ou USER_REOPEN selon la valeur finale.

- *Profile (long)*

L'identificateur du profil auquel est lié le passager.

Utilisée par la passerelle Pass2ldap pour faire la jointure avec les informations maintenues dans le profil.

Mise à jour par l'application PassUr1.

3.5. Les notifications

Les notifications sont créées à l'occurrence d'événements dans l'application PassUr1. Ces notifications sont ensuite utilisées, de manière asynchrone, par la passerelle Pass2ldap. Elles sont modifiées par la passerelle, après traitement, pour indiquer qu'elles ont été prises en compte.

3.5.1. Stockage

Les notifications sont stockées dans la base de données. Elles sont mappées dans la table notification de la base de données PassUr1.

3.5.2. Caractéristiques

Toutes les caractéristiques des notifications sont initialisées à la création d'une notification et ne plus modifiées ensuite, sauf les caractéristiques treated et treatmentDate, qui sont remises à jour lors du traitement des notifications par la passerelle Pass2ldap.

- *Id (long)*

Id est un identifiant unique, incrémenté automatiquement.

- *Type (String)*

Le type d'une notification indique à quoi correspond l'événement. Les valeurs possibles sont :

USER_CREATE : création d'un utilisateur ;

USER_CHANGE : modification des caractéristiques d'un utilisateur (ou du profil auquel est lié l'utilisateur) ;

USER_CLOSE : invalidation d'un compte ;

USER_REOPEN : remise en fonctionnement d'un compte invalidé.

- *User (long)*

Identifiant unique dans la base de données PassUr1 du passager concerné par la notification.

Utilisée par la passerelle Pass2ldap lors du traitement de la notification.

Initialisée par l'application PassUr1.

- *CreationDate (java.sql.Timestamp)*

Date de l'événement notifié.

Champ informatif uniquement utilisé pour trier les notifications.

Initialisée par l'application PassUr1.

- *TreatmentDate (java.sql.Timestamp)*

Date de traitement de l'événement.

Champ informatif uniquement.

Mise à jour lors du traitement de la notification par la passerelle Pass2ldap.

- *Treated (boolean)*

Vrai si la notification a étét traitée, Faux sinon.

Utilisée par la passerelle Pass2ldap pour ne pas traiter deux fois les mêmes événements.

Initialisée à faux, mise à jour à vrai lors du traitement de la notification par la passerelle Pass2ldap.

3.6. Les traces

Les traces (logs) servent à faciliter le suivi des opérations effectuées dans l'application.

Elles sont créées à l'occurrence d'événements dans l'application PassUr1 et la passerelle Pass2ldap.

3.6.1. Stockage

Les traces sont stockées dans la base de données. Elles sont mappées dans la table log de la base de données PassUr1.

3.6.2. Caractéristiques

Toutes les caractéristiques des traces sont initialisées à la création d'une trace et ne sont plus modifiées ensuite.

- *Type (String)*

Le type d'une trace indique à quoi correspond l'événement. Les valeurs possibles seront déterminées ultérieurement.

- *Date (java.sql.Timestamp)*

Date de la trace.

Champ informatif uniquement utilisé pour trier les notifications.

Initialisée par l'application PassUr1.

- *Text (String)*

Un chaîne de caractères expliquant l'événement.

4. L'application PassUr1

L'application PassUr1 est une application CGI, qui peut donc être accédée depuis n'importe quel navigateur web.

Pour mieux s'intégrer dans l'offre logicielle du projet ESUP-Portail, l'application PassUr1 est développée sous la forme d'un canal uPortal. Les utilisateurs sont donc authentifiés par le mécanisme de Single Sign-On CAS (Central Authentication Service).

4.1. Configuration

La configuration de l'application est décrite au format XML dans un fichier. Un exemple de configuration est donné ci-après.

```
<cpassur1-config>

  <admin>
    <email>passur1-admin@listes.univ-rennes1.fr</email>
    <name>Les administrateurs de l'application PassUR1</name>
  </admin>
```

```
  <!-- la liste des composantes -->
  <departments>

    <!-- la composante IFSIC, et ses gestionnaires -->
    <department>
      <id>913</id>
      <name>IFSIC</name>
      <managers>
        <manager>moigne</manager>
        <manager>bmaret</manager>
      </managers>
    </department>

    <!-- la composante CRI, et ses gestionnaires -->
    <department>
      <id>957</id>
      <name>CRI</name>
      <managers>
        <manager>perrigau</manager>
        <manager>bourges</manager>
      </managers>
    </department>

  </departments>

  <!-- la liste des types d'utilisateurs -->
  <user-types>
    <student-types>
      <student-type>ETUDUR1</student-type>
    </student-types>
    <staff-types>
      <staff-type>ENS</staff-type>
      <staff-type>IATOS</staff-type>
      <staff-type>EXT</staff-type>
    </staff-types>
  </user-types>
</cpassur1-config>
```

4.2. Description fonctionnelle

L'application offre deux interfaces, la première correspondant à la gestion des profils, l'autre à la gestion des passagers.

Toutes les pages comportent (en haut des pages) une liste déroulante présentant les composantes dont l'utilisateur connecté est gestionnaire. Cette liste déroulant permet aux gestionnaires de changer la composante courante (et ainsi d'accéder aux profils et aux utilisateurs de cette composante).

4.2.1. Accueil

- *Utilisateurs non autorisés*

Si l'utilisateur n'est gestionnaire d'aucune composante, l'application le lui indique, et l'utilisateur n'a accès à aucune page de l'application.

- *Choix de la composante*

Si l'utilisateur n'a pas choisi sa composante, alors on lui propose la liste des composantes dont il est gestionnaire.

Si l'utilisateur est gestionnaire d'une seule composante, alors la composante est choisie pour lui.

- *Choix de l'interface*

Si l'utilisateur a choisi la composante dont il veut gérer les passagers (ou bien si l'application l'a fait automatiquement pour lui), on lui propose de choisir l'interface qu'il veut utiliser :

Consultation des logs ;

Gestion des profils étudiant ;

Gestion des passagers étudiant ;

Gestion des profils personnel ;

Gestion des passagers personnel.

Les gestionnaires de plusieurs composantes se voient également proposer un lien hypertexte (ou un bouton) leur permettant d'annuler leur choix de composante, ce qui les fera revenir à l'étape précédente.

Les trois premières interfaces sont détaillées ci-après. Les deux dernières interfaces (concernant les personnels) sont bâties exactement comme les interfaces concernant les étudiants.

Toutes les autres pages de l'application proposent des liens hypertextes permettant de revenir en un clic à la page d'accueil (pour pouvoir changer de composante) ou à n'importe laquelle des 5 interfaces ci-dessus.

4.2.2. Consultation des logs

Il s'agit d'une simple page permettant de consulter les logs de l'application, avec les critères suivants :

Afficher les XXX derniers événements ;

Afficher les événements concernant l'uid XXX.

4.2.3. Gestion des profils étudiant

- *Affichage par défaut*

L'affichage par défaut est la liste des profils de la composante courante. Pour chaque profil, des liens permettent :

De modifier ses propriétés ;

De le supprimer, si aucun utilisateur n'appartient au dit profil ;

De basculer en mode « Gestion des passagers », pour le profil correspondant.

Un bouton permet également de créer un nouveau profil.

- *Ajout d'un profil*

Un formulaire est présenté, qui permet d'indiquer les propriétés du nouveau profil.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des profils.

- *Modification d'un profil*

Un formulaire est présenté, qui permet de mettre à jour les propriétés du profil choisi.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des profils.

- *Suppression d'un profil*

Une page de confirmation, sous forme d'un formulaire, est proposée à l'utilisateur pour le prévenir de la destruction du profil.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des profils.

4.2.4. Gestion des passagers

- *Affichage par défaut*

Un lien hypertexte en haut à gauche permet de revenir à l'affichage par défaut de la gestion des profils. Le profil courant est sélectionné dans une liste déroulante en haut à droite du lien hypertexte de retour à la gestion des profils.

L'affichage par défaut est la liste des passagers du profil courant. Pour chaque passager, des liens permettent :

De modifier ses propriétés ;

De l'invalider (ce qui aura pour conséquence le passage dans l'OU peopleoff par db2ldap) ;

De le revalider (re-passage dans l'OU people) ;

De changer le profil auquel il est lié.

Un bouton permet également de créer un nouveau passager.

- *Ajout d'un passager (lié au profil courant)*

Un formulaire est présenté, qui permet d'indiquer les propriétés du nouveau passager.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des passagers.

- *Modification d'un passager*

Un formulaire est présenté, qui permet de mettre à jour les propriétés du passager choisi.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des passagers.

- *Suppression d'un passager*

Une page de confirmation, sous forme d'un formulaire, est proposée à l'utilisateur pour le prévenir de la destruction du passager.

La soumission, ainsi que l'annulation du formulaire, font revenir l'utilisateur à l'affichage par défaut de la gestion des passagers.

5. L'application Pass2ldap

L'application Pass2ldap est une passerelle, opérant depuis la base de données PassUr1 vers l'annuaire LDAP.

Le but de l'application est de répercuter dans l'annuaire LDAP les opérations effectuées dans la base de données PassUr1 à l'aide de l'application du même nom. Pour cela, la passerelle va périodiquement inspecter les nouvelles notifications dans la base de données.

Les notifications non encore traitées sont prises en charge les unes après les autres par ordre chronologique. Toute erreur lors du traitement d'une notification doit immédiatement être signalée aux administrateurs par un mécanisme de remontée d'alertes à préciser (par courrier électronique par exemple).

Cette partie décrit les actions à effectuer en fonction du type des notifications.

5.1. Création d'un passager

La création d'un passager est signalée par une notification de type USER_CREATE.

En réponse à la présence d'une telle notification, l'application :

Choisit un uid non attribué dans l'annuaire LDAP pour le nouvel utilisateur, et inscrit cet uid dans la base de données PassUr1. La manière de choisir le nouvel uid est détaillée ci-après ;

Crée une nouvelle entrée dans l'annuaire LDAP ;

Renseigne les attributs de la nouvelle entrée à l'aide des caractéristiques du passager trouvées dans la base de données. La mise à jour des attributs est exactement semblable à celle effectuée lors de la modification d'un passager.

Lors de la création d'un utilisateur, le choix doit se faire de manière unique. Le plus simple pour cela est de s'appuyer sur les procédures existantes, à savoir :

Adopter une numérotation factice et donner un uid numérique aux étudiants ;

Utiliser le même algorithme que pour la création des uids des utilisateurs de la base Harpège pour les passagers personnels.

L'attribut LDAP ur12sourcecreation d'un nouvel utilisateur issu de l'application PassUr1 est positionné à PASSUR1.

5.2. Modification des caractéristiques d'un passager

La modification d'un passager par l'application PassUr1 est signalée par une notification de type USER_CHANGE.

En réponse à une telle notification, l'application élimine de l'annuaire LDAP tous les attributs de l'entrée correspondante à l'utilisateur qui sont construits à partir des informations de la base de données PassUr1, puis les reconstruit.

Pour chacun de ces attributs, on indique ci-dessous la manière dont ils sont reconstruits. Pour plus de simplicité, on utilise les notations suivantes :

LDAP.x désigne l'attribut x de l'entrée du passager dans l'annuaire LDAP ;

PassUr1.User.y désigne le champ y du passager dans la table user ;

PassUr1.Profile.z désigne le champ z du profil du passager dans la table profile.

Les attributs de l'annuaire LDAP d'un passager sont construits de la manière suivante :

LDAP.cn, LDAP.gecos, LDAP.sn et LDAP.givenname, LDAP.ur1nomusuel, LDAP.ur1nompatronymique, LDAP.ur1prenom, LDAP.ur1nomusuelascii, LDAP.ur1nomascii, LDAP.ur1prenomascii : à partir de PassUr1.User.UsualName, PassUr1.User.GivenName et PassUr1.User.PatronymicName ;

LDAP.nomusuel

LDAP.ur1typeentree est positionné à etu ou pers, selon la valeur de PassUr1.Profile.student ;

LDAP.employeetype à partir de PassUr1.Profile.employeeType ;

LDAP.department à partir de PassUr1.Profile.departmentNumberList ;

LDAP.ur1composante à partir de PassUr1.Profile.ur1ComposanteList

LDAP.ur1inscription à partir de PassUr1.Profile.ur1InscriptionList.

Les champs LDAP.ur1

5.3. Fermeture du compte d'un passager

La fermeture du compte d'un passager est effectuée lorsque l'application trouve une notification de type USER_CLOSE.

Cette procédure existe, elle est donc utilisée telle quelle.

5.4. Ré-ouverture du compte d'un passager

La fermeture du compte d'un passager est effectuée lorsque l'application trouve une notification de type USER_REOPEN.

Cette procédure existe, elle est donc utilisée telle quelle.

5.5. Péremption du compte d'un passager

La péremption d'un compte se fait de manière automatique, sur inspection régulière de l'attribut ur1datefincompte.

La procédure est donc utilisée telle quelle.

Note : faut-il remonter l'information dans la base PassUr1 ?

6. TODO

Préciser les encodages des caractéristiques des passagers (accents, casse).