WWSympa, une interface web pour Sympa

■ Olivier SALAÜN, olivier.salaun@cru.fr
Comité réseau des universités

WWSympa est né du besoin d'une interface web homogène et complète au robot de listes de diffusion Sympa. Cette interface doit donner accès à toutes les fonctions du robot aux utilisateurs, gestionnaires et modérateurs de listes.

■ Sympa, un logiciel serveur de listes de diffusion

Sympa est un robot gestionnaire de listes de diffusion, développé en Perl, fonctionnant sous UNIX, diffusé sous une licence GPL (Gnu Public Licence) ou Artistic. Successeur de Tulp, il est une alternative à Majordomo offrant de bonnes performances et des fonctionnalités originales :

- Internationalisation : Sympa a initialement été pensé pour être multi-lingue. Les dictionnaires de messages sont dissociés du code permettant de choisir la langue utilisée pour les messages renvoyés par le robot. Quatre catalogues (français, anglais, allemand, espagnol) sont livrés avec la distribution.
- Personnalisation : la définition minimale d'une liste est constituée de son nom, son sujet et de l'adresse du propriétaire. Mais les listes peuvent être entièrement personnalisées :
  - Personnalisation du texte des messages de services (charte, désabonnement...) renvoyés par le robot.
  - Contrôle fin des droits de diffusion. Sympa propose plusieurs nuances entre des listes ouvertes et modérées.
  - Possibilité de redéfinir le traitement des commandes grâce à un petit langage de contrôle : les scénarii.
  - Possibilité d'ajout de champs d'en-tête SMTP personnalisés dans les messages diffusés. *Exemple :* List-info : http://listes.cru.fr/wws/info/rsup

Sources des abonnés : les listes d'abonnés peuvent être extraites dynamiquement d'une base de données (MySQL, PostgreSQL, Oracle...) ou d'un annuaire LDAP. Possibilité d'inclusion de listes.

Facilité d'administration : l'intégralité de la configuration d'une liste est concentrée dans 1 seul fichier. Les alias de messagerie sont les même pour toutes les listes et sont indépendants de la configuration de celle-ci.

Sympa a une base de 150 sites installés, principalement des universités françaises, mais aussi des entreprises ou des prestataires de services dans une dizaine de pays. Les dictionnaires chinois et arabe sont en cours de développement.

■ Objectifs de WWSympa

Le CRU (Comité Réseaux des Universités) a développé pour ses besoins propres des outils « maison » destinés aux utilisateurs et gestionnaires de listes. WWSympa concrétise à la fois l'intégration de ces services et le rapprochement du robot de listes Sympa. WWSympa partage notamment les données de Sympa et ne nécessite donc pas une administration séparée.

Avant de se lancer dans le développement de WWSympa, nous nous sommes intéressés à l'existant. Le bilan : beaucoup d'offres de services (egroups, onelist, listbot…) mais peu de produits intégrés et fonctionels.

Fonctionnalités

On distingue différentes populations impliquées dans un service de listes de diffusion : les abonnés, les gestionnaires de listes, les modérateurs et l'administrateur du site (listmaster). WWSympa doit leur donner accès à l'ensemble des fonctions offertes par le robot de listes depuis un navigateur :

- Aux abonnés :
  - abonnement/désabonnement (SUBSCRIBE, SIGNOFF),
  - répertoire des listes (LISTS),

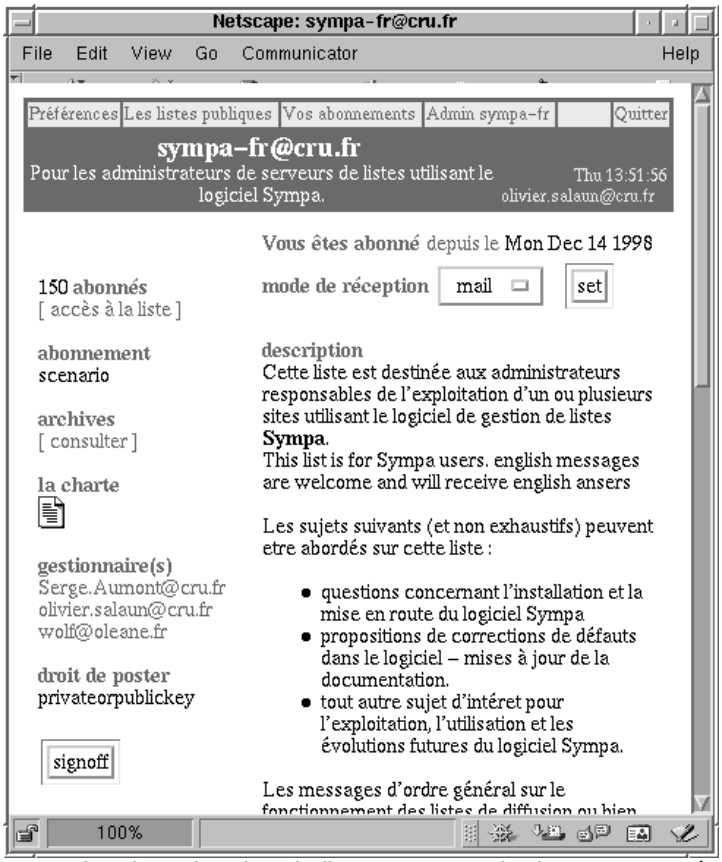
- accès contrôlé aux archives de liste (`INDEX`, `GET`),
- modification des options d'abonnement (`DIGEST`, `CONCEAL`),
- liste des abonnements (`WHICH`).

- Aux gestionnaires :
  - ajout et suppression d'abonnés (`ADD`, `DEL`),
  - accès à la liste des abonnés,
  - expiration des abonnements (`EXPIRE`),
  - spécifiques à l'interface web : paramétrage de la liste, personnalisation des messages de service, gestion des bounces, statistiques,
  - demande de création de liste.
- Aux modérateurs :
  - liste des messages à modérer (`MODINDEX`),
  - diffusion/rejet de messages (`DISTRIBUTE`/`REJECT`).
- Au listmaster :
  - spécifiques à l'interface web : configuration du serveur, validation des demandes de création de liste.

Interface utilisateurs

`WWSympa` propose à chaque utilisateur un environnement personnalisé. Avant identification, l'utilisateur n'a accès qu'aux informations publiques du site (aide + liste des listes publiques). Dès son authentification, l'interface web lui propose toutes les opérations privilégiées (accès aux archives privées, liste de ses abonnements, module d'administration s'il est gestionnaire de liste, module de modération s'il est modérateur, etc.).

L'interface utilisateur de `WWSympa` propose à l'utilisateur une zone de navigation donnant accès aux fonctions



d'authentification ainsi qu'aux fonctions du robot de listes. Le reste du document est réservé aux contenus. Le bandeau de navigation se compose de :

- La barre de navigation donnant accès en permanence aux fonctions « Connexion/Déconnexion », « préférences », « Vos listes ». L'affichage des autres fonctions est déterminé par les privilèges de l'utilisateur et le contexte. *Exemple* : `accès à l'administration de la liste depuis`

l'éditeur du message de bienvenue si l'utilisateur est gestionnaire de la liste.

- 2 niveaux de titres. *Exemple* : « administration de la liste sympa-fr » et « liste des abonnés »
- L'adresse email de l'utilisateur connecté ou une bannière de login si l'utilisateur ne s'est pas authentifié.

■ Relation avec le robot de liste Sympa

Contraintes

WWSympa, dédié au robot de listes Sympa, doit s'interfacer avec ce dernier. Cela induit des contraintes dans l'implémentation de l'application et une analyse des besoins en performances.

L'utilisation de l'API de Sympa, impose d'utiliser Perl pour le développement de WWSympa. Le code de Sympa, modulaire, est composé de plusieurs modules objet pour manipuler les listes, les archives, les logs, le fichier de configuration, les commandes. L'utilisation de cette API par WWSympa assure une compatibilité avec la version actuelle de Sympa et ses évolutions futures.

Par rapport à une interface batch comme celle d'un robot de listes en mode messagerie, le web, service interactif, impose des temps de réponse beaucoup plus faibles. L'utilisateur est en attente d'une réponse immédiate du serveur HTTP. Pour Sympa (donc pour WWSympa), l'opération la plus coûteuse consiste à renvoyer à l'utilisateur l'ensemble des listes auxquelles il est abonné (commande WHICH). En effet, la recherche de cette liste d'abonnements oblige le robot à charger toutes les listes du serveur et à rechercher l'utilisateur concerné dans chacune des listes d'abonnés. Il faut 30 secondes à Sympa pour traiter cette commande sur une base de 80 000 abonnés. Par contre, Sympa est un démon et garde en mémoire les listes d'abonnés, ce qui lui permet de traiter les commandes suivantes en 1 seconde. Un CGI tel que WWSympa, lancé par le serveur HTTP à chaque requête, ne bénéficie pas de cette persistance des données en mémoire.

Le module FastCGI inclus dans le serveur HTTP permet d'exécuter des CGIs persistants. Une instance du CGI est lancé au démarrage du serveur web, puis le CGI reste en attente des requêtes des client HTTP. L'utilisation de FastCGI a permis de résoudre les problèmes de performances de WWSympa. L'utilisation de FastCGI est configurable dans wwsympa.conf.

Partage des données avec le serveur

WWSympa accède aux fichiers de données de Sympa en lecture et en écriture. Il doit en particulier ajouter et supprimer des abonnés de la base des abonnés, ce qui pose un problème de concurrence d'accès à ces données.

Ces nouveaux besoins nous ont amenés à étendre le mode d'accès aux listes d'abonnés dans Sympa. Initialement, la base des abonnés était stockée dans un fichier au format texte, puis les charge dans un BTree en mémoire ; le fichier étant réécrit à chaque opération de mise à jour. Les fonctions d'accès aux données dans Sympa ont été réécrites pour pouvoir déléguer la gestion de la base des abonnés à un SGBD (MySQL ou PostgreSQL). Ce mode de fonctionnement résout le délicat problème de conflit d'écriture et supprime quasiment toutes limites quant au nombre d'abonnés.

La base des abonnés est constituée des 2 tables user et subscriber dont la structure est la suivante :

User
Email
Gecos
Password
cookie_delay
Lang

Subscriber
user
list
date
reception
visibility

L'utilisation du mode de stockage de la liste des abonnés (texte ou SGBD) est choisie pour chacune des listes.

■ Implémentation de WWSympa

Modèle d'authentification

L'authentification des utilisateurs dans WWSympa utilise 2 méthodes :

- authentification initiale : saisie du mot de passe,
- persistance de la connexion : utilisation de cookies.

Authentification initiale

L'authentification initiale est entièrement prise en charge par WWSympa ; cette tâche n'est pas déléguée au serveur HTTP (mécanisme des .htaccess).

- Le mot de passe d'un abonné est un champ dans la base de données partagée avec Sympa. WWSympa effectue des opérations d'ajout, de suppression et de mise à jour des mots de passe. WWSympa n'utilise pas les fichiers de configuration du serveur web. Possibilité de déconnexion de l'utilisateur (confidentialité). Dans le cas d'une authentification par le serveur HTTP, le navigateur garde le couple user/password pour le transmettre au serveur à chaque session et ne l'écrase qu'à la fermeture de l'application. WWSympa permet à l'utilisateur de se déconnecter à tout moment (commande logout).
- Sécurisation des échanges de mots de passe entre le navigateur et le serveur HTTP. WWSympa ne transmet pas le mot de passe en clair dans les cookies (utilisation de MD5).

Les mots de passe sont stockés en clair dans la base de données de Sympa. La clef primaire de la base de données est l'adresse email de l'utilisateur ; c'est l'identifiant qu'il fournit pour s'authentifier.

Cette phase constitue le point faible du système d'authentification de WWSympa puisque le mot de passe saisi par l'utilisateur est transmis au serveur en clair (base64). Plusieurs solutions sont envisageable pour sécuriser la transmission du mot de passe :

- Installation d'un serveur HTTPS (utilisant de SSL pour crypter les données circulant entre le client et le serveur).
- Utilisation d'une classe Java (coté client) pour crypter le mot de passe avant l'échange. Cette méthode est utilisé par Rearsite (gestion de Homedir via le web).

Persistance de la connexion

Le cookie permet d'assurer la persistance de la connexion pour un utilisateur donné depuis un poste de travail en assurant au serveur que le client a bien été authentifié.

A chaque requête, WWSympa vérifie la validité de la clef MD5 en la recalculant à partir du mot de passe trouvé dans la base de données. La date d'expiration du cookie est positionnée à **`t + cookie_delay`** , cookie_delay étant le délai d'expiration paramétrable par chaque utilisateur. Le format du cookie est le suivant :

```
Set_cookie: user=<email>:<clef> ; expire=<date>
<email> : email de l'utilisateur.
<clef> : hachage MD5 des éléments suivants :
1.  adresse email
2.  @ IP du client HTTP
3.  mot de passe crypté.
<date> : date d'expiration du cookie.
```

Le cookie ne fournit pas les privilèges de l'adresse spécifiée. Pour les déterminer, WWSympa consulte la base de données pour chaque opération et découvre si l'utilisateur est abonné, propriétaire, modérateur ou listmaster afin d'en déduire ses privilèges. Ce mode de fonctionnement prévient les risque d'usurpation d'identité pas falsification d'un cookie, mais le cookie est réutilisable depuis la même machine.

L'allocation initiale ou la perte d'un mot de passe ne peut être traitée par une personne (plus de 80 000 abonnés différents dans cru.fr). WWSympa permet donc de recevoir son premier mot de passe ou de se le faire rappeler par email. La solidité du système d'authentification de WWSympa dépend ici de celle du serveur de messagerie de l'utilisateur.

Navigation

L'interface de WWSympa n'utilise pas de cadres (frame) qui induisent certains problèmes de navigation :

- une mauvaise gestion de l'espace disponible : problème de redimentionnement des cadres, trop de fenêtres avec ascenseurs,
- l'impossibilité d'accéder directement au document ; l'utilisateur positionne des signets pointant vers le frameset.

L'utilisation des tableaux HTML dans WWSympa permet de simuler une navigation par frames. La complexité induite par plusieurs niveaux de tableaux reste acceptable puisque l'intégralité de l'interface est définie dans un seul gabarit HTML.

WWSympa est constitué d'un seul CGI. Afin de rendre les URLs lisibles, il manipule ses paramètres dans la variable d'environnement PATH_INFO.

```
   Exemple : wwsympa/editfile/internet-actu/welcome
   appelle wwsympa en lui passant les paramètres :
   1.  action=editfile
   2.  list=internet-actu
   3.  file=welcome
   Soit l'édition du message de bienvenue de la liste inter-
net-actu.
```

Manipulation de la charte graphique

Dans WWSympa, le code et le source HTML sont entièrement séparés. Il est ainsi possible de redéfinir l'aspect des pages sans intervention dans le code. En outre, WWSympa utilise un seul template (gabarit) de page en meta-HTML pour l'ensemble de l'interface. Ce gabarit est interprété en fonction du contexte d'appel. La figure ci-après présente un extrait de ce template.

```
   [IF action=modindex]
        <TABLE><TR>
        <TD>Date</TD><TD>Auteur</TD><TD>Sujet</TD><TD>Taille</TD><TD>Action</TD>
        </TR>
        [FOREACH msg IN spool]
                <TR><TD>
                [IF msg->date]
                        [msg->date]
                [ELSE]
                         
                [ENDIF]
                </TD><TD>[msg->from]</TD> <TD>
                [IF msg->subject]
                        [msg->subject]
                [ELSE]
                         
                [ENDIF]
                </TD><TD>[msg->size] kb </TD><TD>
                <A HREF=»[path_cgi]/viewmod/[list]/[msg->NAME]»>voir</A> /
                <A HREF=»[path_cgi]/distribute/[list]/[msg->NAME]»>distribuer</A> /
                <A HREF=»[path_cgi]/reject/[list]/[msg->NAME]»>rejeter</A>
                </TD></TR>
        [END]
        </TABLE>
[ENDIF]
```

Cette séparation code/données facilite la lisibilité du code et autorise un site utilisateur à personnaliser l'interface utilisateur de WWSympa. Le gabarit doit en outre permettre l'internationalisation du logiciel.

Installation et configuration

WWSympa comprend un CGI, un module Perl, un gabarit HTML et un fichier de configuration. Des catalogues de messages permettront l'internationalisation du produit. La procédure d'installation se limite à personnaliser le fichier de configuration et autoriser l'exécution du CGI dans la hiérarchie web (modification du fichier de configuration du serveur web).

WWSympa partage les fichiers de configuration de Sympa, ce qui simplifie sa configuration. Le seul paramètre indispensable à son fonctionnement est la localisation du fichier sympa.conf. Les autres paramètres définissent des valeurs par défaut, par le délai d'expiration des cookies.

■ Evolutions de WWSympa

De nombreuses évolutions sont envisagées pour les prochaines versions de WWSympa.

Internationalisation

Pour rendre WWSympa multilingue il est nécessaire de personnaliser le gabarit lui-même ainsi que les messages (compte-rendu, erreur). Comme c'est le cas pour le robot Sympa, les catalogues de messages seront stockés dans des fichiers au format XPG4.

Ower vs Super-owner

Actuellement, tous les gestionnaires d'une liste ont les mêmes privilèges, y compris celui de modifier l'autorisation de diffusion (*Exemple* : `passage d'une liste modérée à une liste publique`) ou ajouter un nouveau gestionnaire à la liste. Il apparaît donc nécessaire de définir la notion de gestionnaire privilégié, seul habilité à éditer certains paramètres sensibles d'une liste.

Extension des fonctions listmaster

Au-delà de l'interface utilisateur, `WWSympa` doit offrir des outils de gestion du robot de listes lui-même au listmaster. Ces fonctions d'administration incluent l'arrêt et le redémarrage du serveur ainsi que l'accès aux fichiers de log de `Sympa`.

Edition des scénarii

Le comportement des commandes de `Sympa` sont entièrement configurables par le biais des scénarii.

Exemple : scénario de la commande SUBSCRIBE

```
match (sender, /univ-rennes1.fr/)    SMTP   =>     request_auth
is_owner (sender, etudiants-l)       SMTP   =>     request_auth
is_subscriber (sender, admin-l)             SMTP   =>     do_it
any                                  SMTP   =>     reject
any                                  MD5    =>     do_it
```

Des `scénarii` sont fournis pour l'ensemble des commandes de `Sympa` ; cependant les gestionnaires de listes peuvent définir des `scénarii` personnalisés pour leur liste. Un éditeur de `scénarii` devrait leur faciliter la tâche.

Utilisation des CSS (HTML 4.0)

L'utilisation des feuilles de style (CSS) dans le gabarit HTML de WWSympa en faciliterait la personnalisation. La mise en forme du template occupe actuellement la majeure partie du code HTML (couleur, fonte, tableaux…).

Intégration d'un gestionnaire de bounces

`Anabounce` est une contribution à `Sympa` facilitant la gestion des rapports de non-remise des listes de diffusion. Les `bounces` ne sont plus adressés au gestionnaire de la liste, mais traités par un script, analysés puis stockés. `Anabounce` parvient à retrouver l'adresse incriminée dans 90 % des cas. Le gestionnaire peut ensuite désabonner les abonnés en erreur via une interface web. Cette dernière phase pourrait être entièrement intégrée à `WWSympa`. On peut même envisager l'ajout d'informations sur les `bounces` dans la base du robot `Sympa`.

Classification des listes

Le catalogue des listes fournit par WWSympa se base actuellement sur les paramètres visibility et subject de chaque liste. Le caractère binaire du paramètre visibility (publique / privée) n'autorise aucune classification des listes. L'adjonction d'un paramètre topics à Sympa doit permettre de détailler l'annuaire des listes d'un site.

Exemple de la liste droit-net@cru.fr : `topics informatique/reseaux, droit`

Intégration d'outils de groupware

La communauté des abonnés d'une liste a de forts besoins d'outils de travail collaboratif. Ces outils existent et proposent une interface web ; ils pourraient être intégrés à l'environnement de WWSympa :

- FaqOMatic, un gestionnaire de FAQ (Foire Aux Questions).
- RearSite, une application permettant de gérer des fichiers à partir d'une interface web.