é113

Faire travailler Wims avec Moodle

Georges Khaznadar <georgesk@ofset.org>

association OFSET

juin 2009

Table des matières

1. Échanger avec Wims : langage PHP
   - PHP est le langage de plusieurs ENT libres
   - La bibliothèque `php_raw`
   - Objets spécifiques à l'ENT Moodle
2. Frontières d'une classe virtuelle Wims
   - La granularité des objets d'un ENT
   - Le type `mod/assignment/type/wims`
   - Dilemme de la portée d'une classe Wims dans Moodle
3. Réutilisation d'un exercice Wims sous Moodle
   - Comment on gagne du temps
   - Métadonnées de l'exercice Wims
   - Autres objets Wims réutilisables

Introduction

Cette série de transparents a été réalisée pour un groupe de travail de l'association WimsÉdu, réunissant des personnes qui souhaitent pouvoir intégrer l'usage de WIMS avec celui d'evironnements numériques de travail (ENT). La consultation de ce document peut éventuellement être utile postérieurement, probablement pour voir où en sont les développements, et si certaines idées proposées ici sont implémentées ou non, ou si elles ont été dépassées par d'autres développements.

PHP est le langage de plusieurs ENT libres

Ganesha : un des premiers environnements intégrés
Epistemon : une bifurcation de Ganesha, utilisée à l'ULCO
Iconito : vise les écoles, les collèges et les lycées
Claroline : un des premiers environnements intégrés
Dokeos : une bifurcation de Claroline, faite par le développeur principal
Moodle : un des premiers environnements intégrés cherchant à implémenter l'idée d'ENT

Tous ces environnements ont été implémentés en utilisant un langage PHP et un accès à une base de données MySQL.

La bibliothèque `php_raw`

Toutes les communications entre un service WIMS et un autre programme sont efficacement encadrées par le module WIMS nommé `adm/raw`. L'essentiel de ce module est programmé dans le langage `ModTool`, en utilisant ses primitives documentées, plus quelques primitives non encore documentées pour l'usage ordinaire de `ModTool`. Le module `adm/raw` s'utilise au travers de fonctions nommée `jobs`.
Comme les ENT libres utilisent le langage PHP, il convient de développer une bibliothèque indépendante du fonctionnement particulier de chaque ENT, qui permet de mettre à disposition facilement une interface semblable à celle des `jobs` du module `adm/raw`.

La bibliothèque `php_raw`

J'ai développé en 2007 une telle bibliothèque, d'abord en langage Python puis je l'ai portée en PHP (je suis plus familier de Python). Mes travaux récents sur cette bibliothèque consistent à lui faire émettre des messages d'erreurs plus consistants et à mieux gérer les erreurs de communication quand elles se produisent.
Pour le moment, cette bibliothèque suppose que le service en PHP et le service WIMS soient fonctionnels sur la même machine. On peut probablement faire évoluer ça en développant un peu plus les contrôles sécuritaires à bas niveau entre les services.

Objets spécifiques à l'ENT Moodle

La première implémentation que j'ai réalisée pour lier Moodle et WIMS consistait à définir un module de Moodle qui est une sous-classe de `assignment/type/upload` (?), c'est à dire qui dérive d'un objet gérant des devoirs que les élèves sont censés renvoyer sous forme électronique.
Quand on crée un « devoir de type WIMS », cela fabrique la structure nécessaire au suivi des élèves et de leur notation dans Moodle, et dans l'interface liée au devoir apparaissent des boutons qui permettent d'accéder au service WIMS.

Objets spécifiques à l'ENT Moodle

Pour le professeur : les boutons apparaissent même avant que WIMS n'ait été sollicité. Après un appui sur les boutons, une classe WIMS est créée à la volée

Pour l'étudiant : les boutons n'apparaissent que si la classe existe et qu'elle contient un travail à faire.

Quand on clique un bouton, une nouvelle fenêtre de navigateur s'ouvre, avec deux `frames` : une petite ou invisible, qui est là pour rappeler l'ENT quand on fermera la fenêtre sur WIMS, l'autre pour accéder à WIMS.

La granularité des objets d'un ENT

Chaque ENT peut définir des niveaux d'organisation différents. Ceci n'est pas nécessairement identique à la façon dont WIMS regroupe les choses.
En particulier, WIMS ne permet de donner des exercices à faire aux étudiants que quand ceux-ci sont organisés dans une feuille d'exercices, et une feuille d'exercices contient un ou plusieurs exercices, avec des possibilités de règles particulière : barème, nombre de répétitions imposées, etc.

Le type mod/assignment/type/wims

Dans le module pour Moodle que j'ai écrit, dans la première implémentation, chaque exercice WIMS dans Moodle correspondait à une classe distincte, elle-même censée comporter une seule feuille d'exercices. La feuille d'exercices pouvait contenir un ou plusieurs exercices.
Quand on faisait plus d'une feuille d'exercices dans la même classe, ceux-ci apparaissaient dans un seul objet pour Moodle.
Dans une deuxième implémentation (mi-2008), j'ai plutôt fait en sorte d'associer une classe WIMS unique à une classe Moodle, et multiplier les feuilles d'exercices.

Dilemme de la portée d'une classe Wims dans Moodle

L'avantage d'associer une classe WIMS à une classe Moodle, c'est qu'on peut alors organiser un examen portant sur les exercices des feuilles qui y ont été publiées, sur plus d'une semaine.
L'inconvénient est qu'on ne peut pas facilement dissocier les feuilles de travail.

Dilemme de la portée d'une classe Wims dans Moodle

Par exemple si l'exercice A de Moodle correspond aux feuilles de travail 1 et 2 dans WIMS, et que l'exercice B de Moodle correspond à une feuille de tavail 3, les étudiants voient toutes les feuilles de travail publiées et non cachées dans la même classe WIMS, qu'ils entrent par l'exercice A ou l'exercice B de Moodle.
Le fait de monter/cacher une feuille de travail se gère depuis WIMS, mais pas à l'aide du module `adm/raw` qui permettrait de le faire depuis Moodle.
Si on crée une classe WIMS unique pour un exercice Moodle, ce problème de mélange d'information n'existe plus, mais il devient difficile de gérer les examens et WIMS doit gérer une nombre considérable de classes virtuelles.

Comment on gagne du temps

Un professeur qui utilise WIMS depuis plus d'un an a intérêt à conserver les sources de ses feuilles de travail : celles-ci sont susceptibles de resservir d'une année à l'autre. Rétablir une feuille de travail d'après sa source est l'affaire d'une minute.
À partir de Moodle, on peut gagner à copier/coller des définitions de devoirs d'une année à l'autre. Cependant la définition des objets qui implantent « les exercices WIMS » dans Moodle, si elle peut s'exprimer sous forme d'un texte, n'est pas dans un langage immédiatement accessible au professeur.

Métadonnées de l'exercice Wims

Wims n'est pas actuellement bâti pour communiquer les métadonnées relatives à un exercice sous une forme normalisée (LOM, etc.). Il n'en maintient pas moins quelques métadonnées au sujet des feuilles d'exercice, telles qu'un titre et un texte d'introduction. Ce serait utile de récupérer ces métadonnées, si elles ont été correctement renseignées, pour les attacher aux objets de Moodle bâtis sur une feuille d'exercice WIMS.

Autres objets Wims réutilisables

Les cours et autres documents pas forcément interactifs réalisables avec WIMS gagneraient à être réutilisables dans un ENT sans être recodés.