

Space Details

Key:	ORIOAIworkflow
Name:	ORI-OAI-workflow
Description:	
Creator (Creation Date):	ycolmant@univ-valenciennes.fr (Jul 04, 2008)
Last Modifier (Mod. Date):	ycolmant@univ-valenciennes.fr (Aug 29, 2008)

Available Pages

- Version 1.1 
 - Spécifications
 - Choix techniques
 - Spécifications du module
 - Modélisation
 - Implémentation
 - Installation
 - Changements version 1.1.*
 - Installation via Quick Install
 - Configurations
 - Pré-Requis
 - Fichiers à configurer
 - Autres fichiers importants
 - OsWorkflow
 - Configurations par défaut
 - ANT
 - Shibboleth
 - Utilisation
 - Captures d'écran
 - Aspects pratiques
 - Tests ANT
 - Eclipse
 - Encodage
 - Sauvegardes
 - Sécurisation des Web Services

ORI-OAI-workflow : Gestion du workflow de saisie

Module optionnel

[Voir l'architecture du système](#)

Dans quels cas l'utiliser

Pour mettre en place le workflow du référencement des ressources

Composants obligatoires

- **ORI-OAI-indexing** pour indexer les fiches de métadonnées saisies
- **ORI-OAI-md-editor** pour l'édition des fiches de métadonnées
- **Base de données SQL** pour le stockage des fiches de métadonnées et des données de gestion
- Un **annuaire LDAP** pour la gestion des utilisateurs

Composants optionnels

- **ESUP-serveur-WebDAV** si vous voulez utiliser ce serveur pour y déposer les ressources
- **Canal stockage ESUP Portail** (ou tout autre client WebDAV) pour déposer les ressources sur le serveur ESUP-serveur-WebDAV
- Un **serveur SSO CAS** pour l'authentification (authentification LDAP si non utilisé)

Description

Ce composant est utilisé pour le workflow de référencement des ressources. Les technologies utilisées par ce moteur de publication sont diverses et permettent un paramétrage très fin, complet et puissant en fonction des besoins de l'établissement ou du consortium qui le met en place.

Il sera aussi couplé dans la version 1.5 avec l'outil de gestion des documents dans le système et l'affectation des droits d'accès aux ressources. En attendant cette version, il est possible uniquement de renseigner le lien vers le(s) document(s) depuis l'interface de saisie des métadonnées ; le dépôt devant se faire en dehors du workflow. Dans le cas où vous utilisez le serveur WebDAV pour gérer vos documents, il est donc nécessaire d'utiliser n'importe quel client compatible avec la norme WebDAV en dehors du workflow pour déposer la ressource et copier le lien vers cette ressource dans la fiche de métadonnées. Pour les établissements utilisant l'ENT ESUP Portail, vous avez la possibilité d'utiliser le [canal stockage](#) pour faire le dépôt de la ressource, gérer les droits d'accès (lecture/écriture pour des usagers, groupes d'usagers de votre établissement ou externes en utilisant la fédération d'identités avec shibboleth).

Le moteur de workflow OsWorkflow permet une configuration avancée de toutes les actions effectuées à chacune des étapes de la publication. En effet, par de la configuration, il peut répondre aux besoins des établissements exigeant du déposant la saisie de tous les champs de métadonnées, mais aussi des établissements qui souhaitent une gestion du dépôt plus structurée où la saisie des métadonnées s'effectue en différentes étapes, par différents intervenants. Le formulaire de saisie des métadonnées est supporté par un appel à ORI-OAI-md-editor aux différentes étapes du workflow. Chaque utilisateur aura alors, pour une même fiche, une vue différente. Ceci lui permettra de voir et d'éditer uniquement les métadonnées sur lesquelles il a un droit d'accès particulier.

Les autres technologies utilisées dans ce module sont Acegi Security pour gérer toutes les sécurités d'accès à l'application, une base de données SQL pour le stockage des fiches de métadonnées et les

données de gestion, Hibernate pour rendre les accès à la base de données relationnelle transparents, Compass/Lucène pour réaliser des requêtes rapides et efficaces sur les fiches de métadonnées associées aux données de gestion et enfin JSF pour la gestion des interfaces graphiques.

[Voir la documentation technique](#)

Spécifications

This page last changed on Jan 05, 2009 by ycolmant@univ-valenciennes.fr.

- [Choix techniques](#)
- [Spécifications du module](#)
- [Modélisation](#)
- [Implémentation](#)

Choix techniques

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Choix techniques

Ce paragraphe définit les choix techniques pris après réflexions, tests, etc. Ces choix ont été fonction des technos, de leur efficacité par rapport aux besoins, de leur souplesse mais aussi de leur robustesse, de l'expérience de chacun, des habitudes et technos usités dans la communauté ESUP, etc.

Workflow - OSWorkflow

[OSWorkflow](#) est un produit OpenSource d'OpenSymphony.

OSWorkflow est une bonne solution de Workflow dans le monde de l'OpenSource. On constate qu'il est utilisé dans différents produits et qu'il est choisi par des projets +/- similaires à ORI.

Une de ses principales spécificités est de fonctionner de manière autonome. La servlet d'exemple fourni par défaut et qui tourne sans aucune configuration le montre bien, OSWorkflow peut fonctionner seul. Il est aisé de pluguer OSWorkflow à une application déjà existante. OSWorkflow peut ainsi être vu comme un composant réellement indépendant dans ORI-OAI-Workflow. La possibilité qu'il offre de le commander via de simples appels WEBServices (SOAP) le prouve également.

OSWorkflow gère de manière autonome tout ce qui concerne le Workflow. Les Workflows (diagrammes états-transitions) se configurent via un fichier XML. Une Application Graphique Java peut permettre de créer/modifier des Workflows (attention : cela reste cependant avant tout expérimental). Les Workflows peuvent être relativement complexes. Ils permettent d'appeler des scripts Java BeanShells lors d'une transition (ou mieux encore des méthodes définies dans des "beans spring", et c'est ce qui nous intéresse ici), de faire des splits/joins, de mettre des conditions de tous types sur des transitions (par exemple des conditions sur l'appartenance d'un utilisateur à un rôle), de définir des permissions en fonction des états, etc.

Indépendant, OSWorkflow peut s'utiliser avec sa propre base de données (en tout cas, il gère ses propres tables). L'édition des workflows se fait de manière indépendante du reste de l'application, en éditant le fichier XML OSWorkflow avec un simple éditeur texte. L'indépendance de OSWorkflow ne gêne pas son intégration complète dans le module ORI-OAI-Workflow : on utilise OSWorkflow conjointement à spring, ce qui permet d'utiliser des méthodes spécifiques au module ORI-OAI-Workflow en tant que fonctions et conditions dans des workflows OSWorkflow. Aussi les possibilités du module ORI-OAI-Workflow en matière de conditions ou de fonctions à appeler lors de transitions sont facilement extensibles et cela fait partie des gros points forts de ce module.

Conteneur - uPortal

ORI-OAI-Workflow peut fonctionner dans un ESUP-Portail. Il est décliné sous forme de JSR168. ORI-OAI-Workflow peut également fonctionner sans ESUP, il fonctionne alors de manière "standalone" dans les premières versions, c'est le mode recommandé. Dans ORI-OAI-Workflow, la gestion d'utilisateurs/groupes se fait via un ldap (et donc via un openldap/phpldapadmin local par exemple si le système d'information dans lequel ORI-OAI-Workflow se déploie n'a pas de système de gestion de son ldap). La récupération des utilisateurs et des groupes par ORI-OAI-Workflow sur le ldap est à configurer dans des fichiers de configuration. Il est à noter que ORI-OAI-Workflow est capable de définir des nouveaux groupes "virtuels" via des filtres ldap configurées dans le module ORI-OAI-Workflow (ce qui peut s'avérer extrêmement pratique).

Framework de développement - Spring

[Spring](#) est un framework de développement de haut niveau dans la mesure où il environne et dirige l'architecture générale de l'ensemble de l'application. On notera quelques caractéristiques de Spring qui nous ont fait choisir Spring comme Framework principal de l'application.

Il est le choix de toute la communauté ESUP pour les présents et futurs développements. Il est pensé pour intégrer directement un certain nombre d'autres frameworks comme hibernate (mais aussi OSWorkflow, Compass/Lucène, ...). Il rend les applications souples et paramétrables. Il permet de séparer les tâches de développement via un développement par couche. Il permet d'implémenter des architectures de type Objet modélisable usuellement via UML. Il propose de tirer parti de la programmation par aspect pour la gestion des transactions de BD, via des modules très sophistiqués comme EhCache pour le cache, Acegi pour la sécurité (authentification et autorisation), et enfin directement en insérant du code métier supplémentaire (pour réaliser un outil de statistiques par exemple ...).

JSF

Bien que l'accent concernant l'ergonomie et l'IHM soit d'abord mis sur les formulaires d'édition, on souhaite avoir une IHM efficace tout le long de l'application. [JSF](#) se couple parfaitement avec Spring.

L'implémentation choisie de JSF est [MyFaces](#) qui semble être l'implémentation OpenSource de référence. On utilise cependant d'autres bibliothèques JSF comme [Jenja](#) par exemple.

L'utilisation de JSF (via Esup Commons, voir plus bas) nous permet d'avoir un code unique pour les 2 versions d'ORI-Oai-Workflow : mode portlet et mode standalone.

BD SQL

Le choix a finalement été fait d'utiliser une Base de Données SQL transactionnelle unique pour l'ensemble de l'application. Pour manipuler la base de données depuis l'application, le choix s'est porté sur [Hibernate](#) qui s'intègre bien dans Spring. Il est simple à configurer et à utiliser notamment lorsqu'il n'y a pas de BD pré-existante (c'est à dire lorsque la structuration de la base de données est conçue pour les besoins de l'application développée, comme c'est le cas ici).

Acegi Security

[Acegi Security](#) est la solution en terme de sécurité intégrée dans une application Spring.

Acegi Security est un produit complexe qui couvre un grand nombre d'aspects autour de la sécurité : "authentication " + "authorisation". Il permet surtout de sécuriser en fonction des rôles de l'utilisateur les appels à des méthodes, les urls.

ORI-OAI-Workflow utilise Acegi Security pour : * la gestion de l'authentification.,

- la gestion de RBAC (Role Based Access Control) qui est le Design Pattern décrivant le fait que l'on assigne les permissions à des rôles et non directement aux "Principals" (~ Groupes et Utilisateurs), cf la
 - et donc la gestion des autorisations.
- Ainsi Acegi a été utilisé comme base pour implémenter une solution de RBAC.

Esup Commons

[Esup-Portail Commons](#) permet de mettre en place (et de façon standard par rapport aux développements Esup) une architecture basée sur Spring-Hibernate-JSF (le framework général de l'application étant Spring : conteneur léger), packagée au mieux, pensée pour les mises à jours futures, pouvant tourner à la fois (avec le même code) en mode portlet et en mode standalone.

Esup Commons est, au vue d'une application comme Ori-Oai-Workflow, une proposition d'architecture représentant des facilités de développement et déploiement. Nous utilisons dans Ori-Oai-Workflow certaines fonctionnalités de Esup Commons.

Ainsi la gestion des exceptions et l'utilisation de Hibernate sont gérées via Esup Commons, tout comme les envois de message via SMTP. Par contre l'authentification est par exemple gérée par Acegi Security et non Esup Commons.

Autres ...

D'autres technologies sont utilisées comme XFire pour l'implémentation du protocole SOAP, SpringModules pour l'intégration d'autres technologies encore: par exemple EHCACHE.

Conclusion

ORI-OAI-Workflow est géré par le framework Spring et répond en JSR168 ou mode standalone. Il contient toute la logique applicative. Il fait appel à OSWorkflow pour afficher les informations relatives aux états dans lesquels se trouvent les workflows et pour procéder à des actions sur les états (transitions). Il propose ainsi à l'utilisateur certaines actions sur les entrées de MétaDonnées qu'il liste : faire une transition sur le workflow correspondant à l'entrée, éditer la fiche XML de l'entrée (et donc appeler un formulaire Orbeon Forms), etc. Il intègre également complètement OSWorkflow. OSWorkflow gère tout ce qui est relatif au Workflow a proprement parlé. il contient et propose les informations relatives aux états des workflows, les transitions possibles, les permissions que l'utilisateur courant a par rapport au rôle dans lequel il est, etc. Il appelle la partie principale lors des conditions et fonctions paramétrées dans les workflow.

Spécifications du module

This page last changed on Jan 05, 2009 by ycolmant@univ-valenciennes.fr.

Spécifications

Workflow (OSWorkflow)

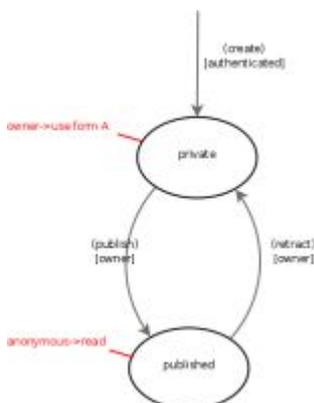
Un Workflow est un concept large. Même si l'on peut être d'accord sur le concept, il peut être décrit de différentes manières, sous différents points de vue (états/transitions, ...) et avec des vocabulaires différents.

Ce paragraphe a donc pour objectif de donner une "vision ORI-OAI" de ce qu'est un workflow. Concrètement, la spécification d'un workflow ORI-OAI correspond au workflow tel qu'il est présenté dans OSWorkflow mais en utilisant en plus des beans spring ORI-OAI comme méthodes à appeler lors des conditions et fonctions du workflow.

Workflow - Diagramme d'état transitions

Un Workflow est défini par son diagramme d'états/transitions. OSWorkflow permet de définir des Workflows par des fichiers XML. Ainsi on définit un Workflow pour la gestion des fiches des ressources pédagogiques, un autre pour la gestion des thèses, etc.

Une fois un Workflow défini, les applications tierces à OSWorkflow peuvent instancier des instances de workflows pour chacune des ressources souhaitées et les manipuler.



Ici est défini un workflow très simple à 2 états : private et publish. On peut passer d'un état à un autre via une transition possible si la condition est respectée. Lors de cette transition, le workflow peut demander à appeler des fonctions.

Dans Ori-Oai-Workflow, un Workflow correspond très concrètement à un fichier de configuration d'un workflow OsWorkflow (implémentant le diagramme ci-dessus par exemple).

Éléments constitutifs d'un Workflow

instance de workflow

Dans Ori-Oai-Workflow, à une entrée XML (~ fiche de Métadonnée XML) correspond une instance de workflow d'un Workflow (diagramme états/transitions) donné. L'instance de workflow permet la gestion du "cycle de vie" de l'entrée XML. L'instance de workflow garde en mémoire les transitions/états passés et présent de l'entrée XML. Elle est consultée pour récupérer des informations comme les transitions disponibles, l'état actuel, les permissions. Elle est aussi appelée pour demander à effectuer les transitions. Elle correspond très concrètement à "une" entrée dans la base de données.

Rôles

Les utilisateurs ont des rôles sur des ressources. Typiquement, un utilisateur aura le rôle de "owner" sur les ressources qu'il a lui-même créées. Les rôles peuvent intervenir dans les conditions : "à condition que l'utilisateur courant soit owner". Ces rôles correspondent en fait concrètement à des entiers qui représentent dans l'application générale les identifications des rôles.

Permissions

Des permissions peuvent être positionnées en fonction de l'état (voir des états) dans lequel se trouve l'instance de workflow. Ces permissions correspondent en fait concrètement à des entiers qui représentent dans l'application générale les identifications des permissions.

Etats-Transitions

Le Workflow est défini par un diagramme d'états/transitions. Une instance de workflow est caractérisée par l'état (voir les états dans le cas de split) dans lequel elle se trouve. Depuis un état, il est possible d'effectuer ou non des transitions. Ces transitions amènent l'instance dans un autre état.

Dans OSWorkflow, on parle de "step" (pour état étape) et de "action" (pour transition).

fonctions - conditions

Les fonctions, tout comme les conditions, peuvent être des java beanshell (scripts en java) ou des classes (méthodes de classes) ou encore, et c'est ce que le déployeur/développeur utilisera certainement la plupart du temps, des beans Spring. Ils peuvent admettre des arguments. Une fonction permet de lancer un script/méthode après ou avant une transition. On peut par exemple y implémenter l'envoi de mail au owner de l'instance du workflow, une demande d'indexation au module Ori-Oai-Indexing ou plus usuellement encore, le positionnement de permissions et/ou de rôles.

Une condition permet de conditionner une transition via un script/méthode. On peut par exemple conditionner une transition sur l'appartenance de l'utilisateur à un rôle donné.

Mise en oeuvre

Comme dit ci-avant, on utilise OSWorkflow comme un sous-module indépendant des autres composants. Aussi, on propose par défaut deux (voir plus) Workflows définis par des fichiers XML. Dans ces Workflows sont spécifiés les permissions, rôles utilisés. Libre aux utilisateurs/déployeurs de ORI de configurer/développer de nouveaux Workflows en prenant comme exemple ceux donnés par défaut.

OSWorkflow est dirigé via Spring et les beans Spring fournis en standard avec OSWorkflow. La persistance de OSWorkflow dans Hibernate est configurée via la configuration des beans Spring OSWorkflow également. Les fonctions/conditions définis dans les workflow peuvent être des beans définis via les fichiers de config spring. [Voir ce lien.](#) Cela nous permet d'intégrer parfaitement OsWorkflow dans l'application principale tout en conservant toute la souplesse d'OsWorkflow. Il est ainsi possible aux utilisateurs de développer et intégrer leurs propres fonctions et conditions dans les workflows

OSWorkflow (et des les partager avec la communauté 😊). C'est donc un excellent point d'entrée pour un développeur désirant étendre les possibilités/fonctionnalités d'Ori-Oai-Workflow.

Configurations

Grâce à Spring mais aussi à OsWorkflow, l'application est souple car paramétrable à différents niveaux. Du même coup, configurer Ori-Oai-Workflow de manière avancée demande d'éditer un certain nombre de fichiers. Les fichiers à configurer/modifier les plus importants (en terme fonctionnel) sont les fichiers de configuration spring, mais aussi les fichiers permettant d'implémenter un workflow OsWorkflow.

Modélisation

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Modélisation

Cette section présente une sorte de première analyse de Ori-Oai-Workflow. Même si par rapport à l'implémentation qui en a été faite, cette modélisation reste très conceptuelle, elle permet de mieux appréhender l'architecture de Ori-Oai-Workflow.

Cas d'Utilisations

Ces différents cas d'utilisation seraient à affiner et compléter.

Pour une étude fonctionnelle poussée (et non pas une approche technique) des cas d'utilisations, référez-vous au document décrivant les cas d'utilisations "modèle" (auteurs: Rosa María Gómez de Regil et Romuald Lorthioir)

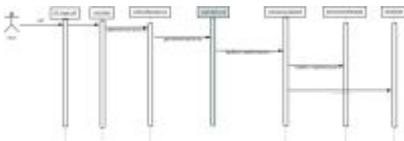


Diagrammes de séquence

Dans les diagrammes de séquence présentés ci-dessous, on constate que OsWorkflow dirige l'application : en fonction de la configuration du workflow actif et de l'action demandé par l'utilisateur ("initialiser un workflow", "valider la publication", "rendre priver", "demander la validation technique", etc), de l'état dans lequel se trouve l'instance de Workflow (donc ~ la fiche de métadonnées), OSWorkflow va appeler les méthodes métiers d'Ori-Oai-Workflow pour vérifier les conditions ou effectuer les fonctions adéquates.

Afficher la liste des entrées.

L'utilisateur demande à afficher les différentes entrées dans lesquelles il peut jouer un rôle sur une instance de workflow.



Créer/Modifier une instance de Workflow

L'utilisateur demande à créer/modifier une fiche de métadonnée et donc une instance de Workflow.

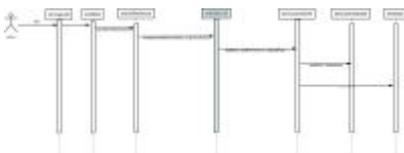


Diagramme de classes

Ce diagramme de classes se veut simple pour une bonne compréhension de l'ensemble. Il ne présente pas un certains nombres d'attributs et surtout de classes qui sont d'ordre architectural (architecture par couche : données/dao/domain/web). De plus certaines associations n'ont pas été implémentées directement comme des associations POJO/Java : ce sont plus des associations conceptuelles établies par les configurations Spring et les services métiers.



Quelques Explications sur certaines classes :

Entry

Correspond à un ensemble d'instances de Workflows. Les instances de Workflows sont dans une même Entrée car elles décrivent la même entité. Elles sont distinctes car elles peuvent correspondrent à des versions différentes, décrivent l'entité avec des schémas différents.

WorkflowInstance

Une instance de Workflow est défini par son contenu XML et son osworkflow (instance d'un OSWorkflow) associé (idworkflow + workflowName).

Un workflowInstance représente donc La donnée principale autour de laquelle gravite l'application Spring.

MetadataType

xpathTitle est le XPath correspondant au titre d'un XML répondant au schéma présent. Les différents MetadataType représentent les différents types de fiches de métadonnées que support l'application. Ils sont à configurer dans un fichier de config spring spécifique (cf la section Configuration de ce présent document).

Concrètement, les différents types de fiches sont les différents types de fiches que pourra ajouter l'utilisateur (en fonction des droits qu'il a et donc des conditions des workflows spécifiés dans le MetadataType) : ** Ajouter une thèse

- ◦ Ajouter une ressource pédagogique
- etc.

RoleStepCategory

Les différents RoleStepCategory représentent les différentes catégories de fiches (~ dossiers virtuels) listées à l'utilisateur. Ils permettent de lister dans un même ensemble des fiches de workflows différentes dans des états différents et sur lesquels l'utilisateur a des rôles différents. Ainsi on peut par exemple présenter à l'utilisateur toutes les fiches dont il est propriétaire qui sont dans l'états creation ou saisie_auteur issues des workflows model1 et easy par exemple.

Ils sont à configurer dans un fichier de config spring spécifique (cf la section Configuration de ce présent document).

Les permissions sont à la fois positionnées par OSWorkflow et par Spring lors de l'initialisation des "permissions par défaut".

Permet d'indexer, rechercher et naviguer plus rapidement dans les fiches. On utilise Compass via Spring en le synchronisant directement sur les transactions Hibernate.

SpringLdap

Ldap sert à l'application de base de données utilisateurs/groupes. De même que pour les autres technos, Spring fournit des beans permettant d'interagir avec Ldap. Ils sont utilisés au travers d'Acegi.

Acegi Security

Acegi Security permet de gérer tout l'aspect security de l'application.

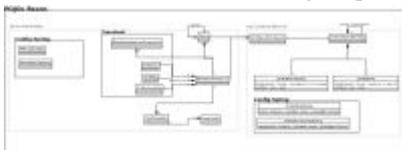
Acegi Security communique avec les sources de données utilisateurs/groupes pour l'authentification ainsi que comme base utilisateurs pour l'implémentation de RBAC (Role Based Access Control).

Acegi Security propose de base une implémentation d'une politique de sécurité de l'application. L'implémentation qui en a été faite en est ici tout autre : * Les données d'Acegi Security sont rendues persistantes via une implémentation Hibernate d'Acegi.

- Nous avons implémenté le modèle RBAC, ainsi par rapport à l'implémentation de base de Acegi nous avons : *** des rôles et des permissions définis sur les objets de type OriAclObjectIdentity, chaque objet étant associé (1/1) à un objet de type OriObjectAclAware (dont héritent WorkflowInstance et Entry)
 - - un système d'héritage d'objets OriAclObjectIdentity permettant de faire hériter les rôles et permissions d'un parent à un fils
 - une possibilité de définir des permissions et rôles sur l'objet racine de l'arbre d'héritage des objets OriAclObjectIdentity, cela via un fichier de configuration Spring
 - un système de mask où un rôle est définie par sa cible (un objet OriAclObjectIdentity), un recipient (le nom d'un groupe ou d'un utilisateur) et un mask (un entier) alors qu'une permission est définie par sa cible (un objet OriAclObjectIdentity), un recipient (le mask d'un rôle) et un mask (un entier)
- Il est possible d'ajouter/supprimer les masks de permissions et rôles disponibles dans l'application
- Une api permet de supprimer, ajouter vérifier un rôle ou une permission sur un objet de type OriAclObjectIdentity (cette api est notamment utilisée via les fonctions/conditions paramétrées dans les workflows OsWorkflow).

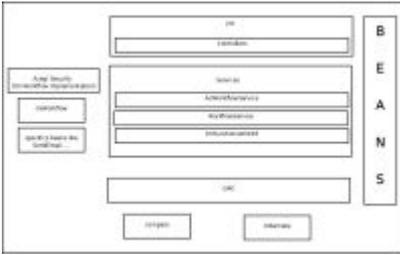
Beans/Pojos Ori-Oai-Workflow-Spring

Le diagramme ci-dessous donne le diagramme de classes des beans/pojos utilisés dans Ori-Oai-Workflow-Spring. Il ne donne pas les beans/pojos issus de OsWorkflow.



- La plupart des beans ainsi que la plupart de leurs attributs sont persistants via Hibernate, ils servent aussi dans le même temps comme DTOs : en fait c'est eux qui sont chargés d'information tout au long des couches Dao et Service puis qui sont directement présentés voir modifiés depuis les contrôleurs webs (cf la section "Architecture par couches" plus bas).
- Certains beans/attributs sont persistants dans les configs Spring uniquement, ils permettent de paramétrer l'application.
- Enfin d'autres sont simplement transients.

Architecture par couches



Comme dit plus haut, les beans (package `org.ori.workflow.beans`, qui sont en fait des `pojos`) sont transversaux aux couches de l'application. Ils représentent à la fois les données persistantes, les données récoltées via la couche service et les données visualisées voir éditées au niveau de la couche Web.

* L'intérêt évident est que les différentes couches s'appuient ainsi sur les mêmes types de données, l'ensemble du code de l'application est ainsi très flexible et léger : l'ajout d'une donnée en tant qu'attribut de bean de la persistance à sa représentation côté web requiert par exemple des modifications très restreintes.

- Le fait cependant d'utiliser les mêmes beans au travers des différentes couches peut cependant, si on n'y prend garde, peut impliquer un certain nombre de contraintes/inconvénients. Cela implique par exemple que les attributs d'un même bean peuvent avoir des persistences différentes : persistance via Hibernate, OsWorkflow ou encore attribut transient ... ce qui peut perturber un développeur découvrant le code. Les objets de type `WorkflowInstance` en sont un exemple parfait. Afin que le développeur s'y retrouve au mieux, il est donc important que les commentaires `JavaDoc` sur les attributs des beans soient le plus précis possible sur le mode de persistance utilisé. De même les beans doivent répondre à la fois aux contraintes données par leur utilisation dans JSF et dans Hibernate. Mais là encore, il est intéressant de constater que l'utilisation du même type de bean à la fois dans Hibernate et dans JSF permet par exemple une implémentation simple d'un `Converter JSF` (voir même d'un `converter générique`, comme cela a été fait pour `ORI-OAI-Workflow`).

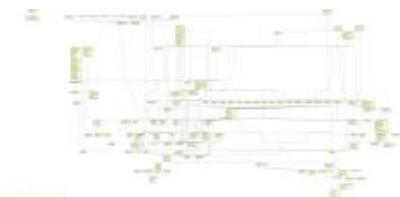
Tout comme le préconise `Esup-Commons`, tous les beans `spring` (on ne parle plus ici des beans `pojos` mais bien des beans `spring`) sont configurés via les fichiers de configuration `spring`. Cela permet d'uniformiser la déclaration des différents beans `spring` et de tirer partie au mieux de `Spring IDE` (dans `Eclipse`).

Au niveau de la couche de présentation dirigée par JSF, on associe à chaque page jsf un `controller`. Ce `controller` permet de récupérer et sauver les beans/`pojos` de l'application. Le choix a été pris de configurer les `controllers JSF` comme des beans `spring` de `scope request`. Un contexte applicatif est cependant maintenu en sauvant les états des beans/`pojos` d'une requête à une autre via la balise `t:savestate` de `MyFaces Tomahawk`. De même le bean/`pojo` `UserContext` par exemple est de `scope session` (mais tous les `controllers` sont bien quant à eux de `scope request`).

La partie services se compose de plusieurs sous parties. * La partie située dans le package `org.ori.workflow.services.acls` est spécifique à la sécurité de l'application. La majeure partie de ce package (qu'il faut mettre en relation avec le package `org.ori.workflow.beans.acls`) correspond à l'implémentation d'`Acegi Security`. `AclWorkflowService` est la classe instanciée pour correspondre au bean `spring` à mettre en relation avec le reste de l'application `ori-oai-workflow`.

- `WorkflowService` appelle à la fois la partie DAO et `OsWorkflow`.
- `OriFonctions4OSWF` ainsi que l'ensemble du package `org.ori.workflow.services.oswfhelper` correspondent aux beans appelés par `OsWorkflow` lors des fonctions et conditions paramétrés dans les workflows.

La partie DAO utilise `hibernate` pour la persistance de données. A noter que la session `hibernate` est gérée (ouverture/fermeture) par `Esup Commons` : on utilise en effet l'implémentation de `Servlet/Portlet` fournie par `Esup Commons`.



Installation

This page last changed on Jan 05, 2009 by ycolmant@univ-valenciennes.fr.

Introduction

Comme dit dans les spécifications les différents modules utilisés dans Ori-Oai-Workflow ainsi que son implémentation via Spring lui confèrent une certaine souplesse. Cette souplesse implique de fait une certaine complexité dans la configuration de l'ensemble de l'application.

- [Changements version 1.1.*](#)
- [Installation via Quick Install](#)
- [Configurations](#)
 - [Pré-Requis](#)
 - [Fichiers à configurer](#)
 - [Autres fichiers importants](#)
 - [OsWorkflow](#)
 - [Configurations par défaut](#)
- [ANT](#)
- [Shibboleth](#)

Changements version 1.1.*

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Changements version 1.1.*

Les valeurs des propriétés principales par défaut ont été modifiées : elles apparaissent entre crochet et en majuscule. Elles peuvent soit être éditées manuellement, soit être modifiées via la configuration du fichier principale du QuickInstall ([Installation via Quick Install](#)).

ori-oai-workflow(-spring)

- Cette version n'utilise plus de base eXist, elle utilise maintenant uniquement la base de données SQL pour le stockage des données.
- Une indexation Compass/Lucène est utilisée, cela permet : *** de supporter sans perte de performance un nombre plus importants de fiches que précédemment : l'accès aux fiches se fait par le biais de requêtes sur l'index lucène.
 - - d'améliorer le moteur de recherche interne dans les fiches de métadonnées
 - de permettre de trier par titre, date, créateur les fiches
- Un certain nombre de fonctionnalités dans ont été ajoutées : *** importation de fiches XML depuis l'IHM
 - - copier/coller/suppression de fiches ou de patrons de fiches (support de plusieurs patrons de fiches)
 - bulles d'aide (que l'on souhaite améliorer prochainement cependant)
 - un certain nombre de méthodes WS ont été ajoutées, cela dans le but d'interactions avec des applications extérieures à ORI-OAI (LMS, CMS, Chaînes Editoriales ...).
- Des améliorations autres ont été apportées (debugage ou consolidation de fonctionnalités ...).

Installation via Quick Install

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

ORI-OAI-commons-quick-install : Installation rapide de ORI-OAI

 **Composant fortement recommandé pour l'installation de ORI-OAI**

Description

Depuis la version 1.1 de ORI-OAI, ce composant permet une installation et une prise en main beaucoup plus rapide du projet avec une configuration centralisée des paramètres de configuration et de déploiement les plus importants. Le principe est de n'éditer qu'un seul fichier de configuration qui est partagé par tous les modules de ORI-OAI. La mise en place de l'outil se fait donc plus rapidement sans avoir à ouvrir et à connaître le fonctionnement de chaque fichier de configuration.

[Voir la documentation technique](#)

Ce mode d'installation permet une installation rapide de l'ensemble en saisissant tous les paramètres généraux aux modules dans un seul fichier de configuration. Vous pouvez toutefois compléter votre installation et vos paramétrages par la suite sans aucun soucis.

Pour ORI-OAI-Workflow :

La documentation d'ori-oai-quick-install peut également être utile pour connaître les paramètres/configurations nécessaires à une installation basique.

Une fois cette installation faite, la documentation ci-dessous devrait vous permettre d'appréhender au mieux l'exploitation d'ori-oai-workflow(-spring).

Configurations

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Configurations

- [Pré-Requis](#)
- [Fichiers à configurer](#)
- [Autres fichiers importants](#)
- [OsWorkflow](#)
- [Configurations par défaut](#)

Pré-Requis

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Pré-Requis

Quel que soit le workflow que vous configurerez, Ori-Oai-Workflow nécessite pré-requis plusieurs services de disponibles : * Une base de données SQL supportant les transactions : MySql en InnoDB ou PostgreSQL (qui a le mérite d'avoir un moteur transactionnel par défaut).

i Dans le cas de MySQL, faites attention à ce que votre base de données pour Ori-Oai-Workflow ait bien le engine storage positionné à innodb !

Sous linux, dans my.cnf, pour les versions de mysql récente, on pourra vérifier que l'on a :

```
default-storage-engine = innodb
```

Pour une version plus anciennes, l'option est de la forme :

```
default-table-type = innodb
```

Commandes pour vérifier en ligne de commande (client mysql) le storage utilisé sur une table de votre base :

```
mysql> use `ori-oai-workflow`;
Database changed
mysql> show table status like 'ORI_WORKFLOW_ACL_ROLE';
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+
| Name                | Type    | Row_format | Rows | Avg_row_length | Data_length
| Max_data_length | Index_length | Data_free | Auto_increment | Create_time |
Update_time | Check_time | Create_options | Comment
|
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+
| ORI_WORKFLOW_ACL_ROLE | InnoDB | Dynamic    | 5 | 3276 | 16384 |
| NULL | 16384 | 0 | 6 | NULL | NULL
| NULL | | InnoDB free: 3072 kB; (`oriAclObjectIdentityId`)
REFER `ori-oai-workflow/ORI_WORKFLOW_ACL_OBJECT_IDENTITY`(`oriAclObjectIdentityId`)
|
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+
1 row in set (0.00 sec)
```

- Un annuaire Ldap dans lequel on va pouvoir récupérer les informations des utilisateurs. Si on utilise l'authentification CAS et non Ldap, il faudra s'assurer que le login CAS correspond bien à l'UID dans Ldap.
- un serveur smtp afin d'envoyer les emails d'exception à un email donné (en fait, peut être optionnel en modifiant le fichier de configuration exceptionHandling.xml).
Le serveur smtp est utilisé également dans les workflows lors de l'envoi de mails aux modérateurs, etc. bien sûr

A cela, et suivant votre workflow, vous aurez certainement besoin d'autres services d'activés comme * le module d'indexation utilisé usuellement lors d'une transition de type "publication".

- ...

Fichiers à configurer

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Fichiers à configurer par l'utilisateur déployant Ori-Oai-Workflow

build.properties

deploy.home permet de définir le répertoire dans lequel on veut déployer l'application

La propriété custom.recover.files permet de lister des fichiers à récupérer lors d'une mise à jour d'une version du module. Notez que dans le build.xml, un certain nombre de fichiers sont de facto récupérés si disponibles dans une version antérieure lors des mises à jour. Actuellement c'est ceux-ci: * conf/properties/main-config.properties

- conf/properties/spring/spring-categories.xml
- conf/properties/spring/spring-metadata-types.xml
- conf/properties/spring/spring-webservices-client.xml
- conf/properties/spring/acegi/acegi-acls-root.xml
- conf/properties/spring/acegi/acegi-authentication-additional-groups.xml
- conf/properties/spring/osworkflow/workflows.xml
- WebContent/WEB-INF/web.xml



Ces fichiers représentant les fichiers que vous aurez sans doute besoin de modifier pour paramétrer vos workflows

Enfin il est possible également de définir un zip à importer dans l'application via un script ant.

conf/properties/main-config.properties

Ce fichier permet de définir les principales configurations d'ordre technique de l'application : authentification ldap/cas, connection ldap, BD SQL, Module d'indexation, serveur smtp, ...

A noter que pour le ldap, vous pouvez "simuler" des groupes via des filtres ldap en configurant le fichier conf/properties/spring/acegi/acegi-authentication-additional-groups.xml

Enfin notez surtout que ce fichier conf/properties/main-config.properties est le fichier principal des configurations et vous renvoie à d'autres fichiers de configuration: les commentaires sont donc à prendre en compte cela vaut pour les autres fichiers de configuration également bien sûr. Il est avant tout destiné à une installation rapide de l'application mais ne permet quasiment pas de personnaliser votre module. Très vite après une première installation rapide, vous aurez certainement besoin de personnaliser et adapter les workflows d'exemple à vos utilisateurs, à vos cas d'utilisations, à votre SI, à votre mode de fonctionnement, ...

properties/spring/spring-categories.xml

Ce fichier permet de paramétrer les catégories de fiches qui seront affichées à l'utilisateur. Cela permet de mettre dans un même ensemble des fiches provenant de différents workflows ayant des états différents, sur lesquelles l'utilisateur a des rôles différents. Ce fichier est donc à modifier en fonction des workflows, permissions et rôles configurés dans l'application et en fonction de ses besoins propres donc.

properties/spring/spring-metadata-types.xml

Ce fichier permet de paramétrer les types de fiches que peut ajouter un utilisateur. Notez toute fois qu'un utilisateur pourra effectivement ajouter une fiche d'un type donné si les conditions d'ajout déclarées dans le workflow correspondant sont vérifiées (conditions sur initial-action).

Un type de fiche Ori-Oai-Workflow est ainsi déterminé par un workflow OsWorkflow (identifié par workflowName), un nom de schéma, un xsl (permettant à partir d'un xml de produire un html), un fichier xml de défaut qui sera donné comme fiche vide à la création d'une nouvelle fiche (sauf dans le cas où l'utilisateur utilise des patrons de fiches).

C'est ici également que l'on configure les XForms à mettre en relation avec les fiches. On a une liste de XForms par type de contenu. Pour chaque XForms est donné un identifiant (réutilisé dans le positionnement des permissions) et une url correspondant à l'adresse du Serveur Orbeon présentant le XForms voulu (et c'est ici par exemple que vous pourrez ajouté un pointeur sur vos formulaires personnalisés ou vous permettant de supporter d'autres formats de MétaDonnées).

Acegi Security

Pour infos, les configurations possibles ici proviennent de l'implémentation spécifique de la politique de sécurité de Ori-Oai-Workflow.

[properties/spring/acegi/acegi-acls-root.xml](#)

Le bean entriesRoot permet de configurer les permissions et rôles par défaut dans l'application.

Les masks utilisés doivent être déclarés dans acegi-permissions.xml

Comme décrit précédemment entriesRoot est utilisé comme racine des objets Ori-Oai-Object-Identity

[properties/spring/acegi/acegi-permissions.xml](#)

Permet de configurer les permissions et rôles disponibles dans l'application.

Notez qu'il faut utiliser des puissances de 2 dans vos définitions de mask. Cela permet pour les permissions (pour l'instant) de simplifier l'écriture des conditions dans les configurations des workflows OsWorkflow.

Ces valeurs définissant les rôles et permissions sont utilisés dans les autres fichiers de configuration. On recommande donc lors de la configuration du module ORI-OAI-Workflow d'avoir un tableau comme celui ci-dessous sous les yeux.

PermissionsMask

Permission	Code
ADMINISTRATION	1
READ	2
WRITE	4
CREATE	8
DELETE	16
MODERATE	32
USE_CREATE_FORM	64
USE_LOM_FORM	128
USE_SCD_FORM	256
USE_JURIDIQUE_FORM	512
USE_TECHNIQUE_FORM	1024
USE_ENTITY_FORM	2048

RolesMask

Rôle	Code
AUTHENTICATED	1
OWNER	2
MODERATOR	4
ENTITE_MODERATEUR	8
JURIDIQUE_MODERATEUR	16
TECHNIQUE_MODERATEUR	32
SCD_MODERATEUR	64
ADMINISTRATOR	128

Ce tableau correspond aux rôles/permissions donnés par défaut dans une distribution ori-oai-workflow. Vous pouvez le modifier selon vos besoins.

Autres fichiers importants

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Autres fichiers de configuration importants

properties/spring/spring-osworkflow-helpers.xml

C'est ici que sont déclarés les beans spring utilisés dans les workflows OsWorkflow pour implémenter des conditions et des fonctions OsWorkflow.

L'administrateur/développeur voulant rajouter des fonctionnalités dans son Workflow liées à son contexte de déploiement d'Ori pourra de la même façon qu'ici ajouter ses propres beans spring qu'il utilisera dans la configuration des workflows d'OsWorkflow. Le mieux sera de déclarer ses beans dans un fichier dédié à cela (vide par défaut) : conf/properties/spring/spring-custom.xml

OsWorkflow

La configuration de workflow(s) OsWorkflow est un élément capital dans le déploiement d'un Ori-Oai-Workflow. Comme dit ci-avant, c'est le workflow qui dirige le module. Dans Ori-Oai-Workflow, OsWorkflow n'a pas été modifié ou personnalisé, les fichiers de configuration des workflows sont donc des fichiers spécifiques OsWorkflow. Ainsi vous pouvez tout à fait vous référer à la documentation d'OsWorkflow si vous en sentez le besoin de même que pour tous les fichiers de configuration de type Spring, vous pouvez vous référer à la documentation de Spring pour connaître la syntaxe utilisée .

Dans un premier temps, il est important de comprendre ce qu'est un workflow, revoyez dans la section "Specifications" la partie sur le Workflow si ce n'est pas clair.

Dans un second temps, il vous faut connaître les possibilités en matière de condition et fonctions et arriver à penser vos cas d'utilisations du module en terme de Workflow. Pour ce faire, rien ne vaut la présentation d'un exemple exhaustif que vous devrez mettre en relation avec le document fonctionnel sur les cas d'utilisations (document rédigé par Rosa María Gómez de Regil et Romuald Lorthioir).



On notera que ce graphe représentant le workflow des documents que l'on veut indexer n'a pas grand chose de spécifique à OsWorkflow ou à ORI-OAI. Il faut également se rappeler que les rôles, les permissions (dont les permissions d'utiliser tel ou tel formulaire) sont des éléments configurables comme on l'a décrit ci-avant.

Une fois ce graphe conçu, la plus grosse partie du travail est faite : il reste à concevoir le XML implémentant ce diagramme en "langage" OsWorkflow, même si il en résulte un XML qui peut être conséquent, le langage XML d'OsWorkflow n'est pas complexe.

 Ne pas oublier que Ori-Oai-Workflow doit utiliser plusieurs workflows suivant la nature des fiches à indexer, etc. (cf les "MetadataType").

Notes sur les conditions et fonctions disponibles actuellement et en natif dans Ori-Oai-Workflow

Conditions:

HasPermission

Condition sur le fait que l'utilisateur courant a une permission donnée.

HasRole

Condition sur le fait que l'utilisateur courant a un rôle donné.

VerifyXPathes

Vérifie que la résolution de chaque XPath donné en paramètre correspond à au moins un noeud à chaque fois. Place un message dans le listing d'informations de l'instance/fiche considéré si ce n'est pas le cas.

Fonctions:

AddPermission

Ajoute une permission à un rôle sur l'instance courante.

AddRole

Ajoute un rôle à un utilisateur ou groupe ou utilisateur courant sur l'instance courante. Un XPath et une valeur attendue sur le xpath donnée sont des paramètres optionnelles à cette fonction. Ils permettent de conditionner ce positionnement de rôle selon la présence ou non d'une valeur à un xpath donné. Par exemple cela permet de donner le rôle de modérateur au groupe de professeurs de mathématiques si la discipline dans la fiche LOM a été positionnée à Mathématiques.

DeleteIndex

Demande à un ORI-OAI-Indexing de supprimer l'index courant de l'instance courante (si elle a été indexée préalablement).

DeletePermission

Supprime une permission à un rôle sur l'instance courante.

DeleteRole

Supprime un rôle à un utilisateur ou groupe ou utilisateur courant sur l'instance courante.

RevertXml

Restore une version de la fiche d'un ancien état (il faut pour cela que dans cet ancien état, un SaveXmlHistory ait été fait).

SaveOrUpdateIndex

Demande à un ORI-OAI-Indexing d'indexer ou de réindexer la fiche de l'instance courante.



Dans le même temps cette fonction appelle implicitement la fonction SaveXmlHistory.

Cette fonction DOIT obligatoirement être configurée comme étant une post-fonction dans OsWorkflow.

SaveXmlHistory

Sauve le XML actuel comme un historique (archive) de l'état initial à la transition.



Cette fonction DOIT obligatoirement être configurée comme étant une post-fonction dans OsWorkflow.

SendEmail

Envoie des emails aux utilisateurs d'un rôle donné sur l'instance et/ou à un mail donné "en dur" dans le workflow.

Cette fonction est donc capable de récupérer tous les emails des utilisateurs qui ont un rôle via un positionnement direct sur l'instance, via un positionnement au travers d'un groupe, etc. ... cela signifie que le nombre d'emails peut suivant les configurations être conséquent. La récolte des emails peut actuellement prendre un certain temps si la liste est longue (multiples requêtes ldap). [l'envoi des emails se fait toutefois dans des threads séparés]. Nous recommandons la mesure du possible d'utiliser une liste de diffusion (mail "en dur") pour l'envoi de mails à tout un groupe donné et bien défini ...

SetInstanceIdentity et SetEntryIdentity

Place des identifiants (propre à l'instance/fiche ou à l'entrée/entité considéré) au niveau d'un xpath donné.

Notez que l'ordre des fonctions dans le fichier de config n'est pas annodin. Un certain nombre des fonctions présentées ci-dessus vont fonctionner dans un contexte transactionnel mais pas toutes Les fonctions qui tournent dans un contexte transactionnel annuleront leurs effets automatiquement si une des fonctions provoquait une erreur innatendue. Dans ce contexte, et par rapport au listing ci-dessus, il est recommandé de lancer * en tout dernier l'envoi d'emails (notez que contrairement aux autres fonctions, si l'envoi d'email échoue, rien n'est annulé pour autant),

- en avant dernier l'appel à des WebServices et donc à ori-oai-indexing par exemple,

OsWorkflow

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

OsWorkflow

La configuration de workflow(s) OsWorkflow est un élément capital dans le déploiement d'un Ori-Oai-Workflow. Comme dit ci-avant, c'est le workflow qui dirige le module. Dans Ori-Oai-Workflow, OsWorkflow n'a pas été modifié ou personnalisé, les fichiers de configuration des workflows sont donc des fichiers spécifiques OsWorkflow. Ainsi vous pouvez tout à fait vous référer à la documentation d'OsWorkflow si vous en sentez le besoin de même que pour tous les fichiers de configuration de type Spring, vous pouvez vous référer à la documentation de Spring pour connaître la syntaxe utilisée .

Dans un premier temps, il est important de comprendre ce qu'est un workflow, revoyez dans la section "Specifications" la partie sur le Workflow si ce n'est pas clair.

Dans un second temps, il vous faut connaître les possibilités en matière de condition et fonctions et arriver à penser vos cas d'utilisations du module en terme de Workflow. Pour ce faire, rien ne vaut la présentation d'un exemple exhaustif que vous devrez mettre en relation avec le document fonctionnel sur les cas d'utilisations (document rédigé par Rosa María Gómez de Regil et Romuald Lorthioir).



On notera que ce graphe représentant le workflow des documents que l'on veut indexer n'a pas grand chose de spécifique à OsWorkflow ou à ORI-OAI. Il faut également se rappeler que les rôles, les permissions (dont les permissions d'utiliser tel ou tel formulaire) sont des éléments configurables comme on l'a décrit ci-avant.

Une fois ce graphe conçu, la plus grosse partie du travail est faite : il reste à concevoir le XML implémentant ce diagramme en "langage" OsWorkflow, même si il en résulte un XML qui peut être conséquent, le langage XML d'OsWorkflow n'est pas complexe.



Ne pas oublier que Ori-Oai-Workflow doit utiliser plusieurs workflows suivant la nature des fiches à indexer, etc. (cf les "MetadataType").

Notes sur les conditions et fonctions disponibles actuellement et en natif dans Ori-Oai-Workflow

Conditions:

HasPermission

Condition sur le fait que l'utilisateur courant a une permission donnée.

HasRole

Condition sur le fait que l'utilisateur courant a un rôle donné.

VerifyXPathes

Vérifie que la résolution de chaque XPath donné en paramètre correspond à au moins un noeud à chaque fois. Place un message dans le listing d'informations de l'instance/fiche considéré si ce n'est pas le cas.

Fonctions:

AddPermission

Ajoute une permission à un rôle sur l'instance courante.

AddRole

Ajoute un rôle à un utilisateur ou groupe ou utilisateur courant sur l'instance courante. Un XPath et une valeur attendue sur le xpath donnée sont des paramètres optionnelles à cette fonction. Ils permettent de conditionner ce positionnement de rôle selon la présence ou non

d'une valeur à un xpath donné. Par exemple cela permet de donner le rôle de modérateur au groupe de professeurs de mathématiques si la discipline dans la fiche LOM a été positionnée à Mathématiques.

DeleteIndex

Demande à un ORI-OAI-Indexing de supprimer l'index courant de l'instance courante (si elle a été indexée préalablement).

DeletePermission

Supprime une permission à un rôle sur l'instance courante.

DeleteRole

Supprime un rôle à un utilisateur ou groupe ou utilisateur courant sur l'instance courante.

RevertXml

Restore une version de la fiche d'un ancien état (il faut pour cela que dans cet ancien état, un SaveXmlHistory ait été fait).

SaveOrUpdateIndex

Demande à un ORI-OAI-Indexing d'indexer ou de réindexer la fiche de l'instance courante.



Dans le même temps cette fonction appelle implicitement la fonction SaveXmlHistory.

Cette fonction DOIT obligatoirement être configurée comme étant une post-fonction dans OsWorkflow.

SaveXmlHistory

Sauve le XML actuel comme un historique (archive) de l'état initial à la transition.



Cette fonction DOIT obligatoirement être configurée comme étant une post-fonction dans OsWorkflow.

SendEmail

Envoie des emails aux utilisateurs d'un rôle donné sur l'instance et/ou à un mail donné "en dur" dans le workflow.

Cette fonction est donc capable de récupérer tous les emails des utilisateurs qui ont un rôle via un positionnement direct sur l'instance, via un positionnement au travers d'un groupe, etc. ... cela signifie que le nombre d'emails peut suivant les configurations être conséquent. La récolte des emails peut actuellement prendre un certain temps si la liste est longue (multiples requêtes ldap). [l'envoi des emails se fait toutefois dans des threads séparés]. Nous recommandons la mesure du possible d'utiliser une liste de diffusion (mail "en dur") pour l'envoi de mails à tout un groupe donné et bien défini ...

SetInstanceIdentity et SetEntryIdentity

Place des identifiants (propre à l'instance/fiche ou à l'entrée/entité considéré) au niveau d'un xpath donné.

Notez que l'ordre des fonctions dans le fichier de config n'est pas anodin. Un certain nombre des fonctions présentées ci-dessus vont fonctionner dans un contexte transactionnel ... mais pas toutes ... Les fonctions qui tournent dans un contexte transactionnel annuleront leurs effets automatiquement si une des fonctions provoquait une erreur inattendue. Dans ce contexte, et par rapport au listing ci-dessus, il est recommandé de lancer * en tout dernier l'envoi d'emails (notez que contrairement aux autres fonctions, si l'envoi d'email échoue, rien n'est annulé pour autant),

- en avant dernier l'appel à des WebServices et donc à ori-oai-indexing par exemple,


```

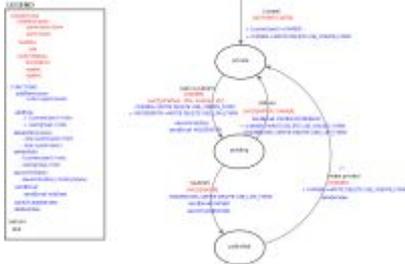
<action id="1" name="Publish">
  <restrict-to>
    <conditions type="AND">
      <condition type="spring">
        <arg name="bean.name">hasRole</arg>
        <arg name="mask">2</arg><!-- Role 2 = OWNER -->
      </condition>
    </conditions>
  </restrict-to>
  <results>
    <unconditional-result status="Underway"
      old-status="Finished" step="2" />
  </results>
  <post-functions>
    <function type="spring">
      <arg name="bean.name">deletePermission</arg>
      <arg name="mask">20</arg><!-- Permission 20 = WRITE + DELETE -->
      <arg name="recipient">2</arg><!-- Role 2 = OWNER -->
    </function>
    <function type="spring">
      <arg name="bean.name">saveOrUpdateIndex</arg>
      <arg name="idOriIndexing">indexingServicePublic</arg>
    </function>
  </post-functions>
</action>
</actions>
</step>
<step id="2" name="Public">
  <actions>
    <action id="2" name="Make Private">
      <restrict-to>
        <conditions type="AND">
          <condition type="spring">
            <arg name="bean.name">hasRole</arg>
            <arg name="mask">2</arg><!-- Role 2 = OWNER -->
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result old-status="Finished"
          status="Underway" step="1" />
      </results>
      <post-functions>
        <function type="spring">
          <arg name="bean.name">addPermission</arg>
          <arg name="mask">20</arg><!-- Permission 20 = WRITE + DELETE -->
          <arg name="recipient">2</arg><!-- Role 2 = OWNER -->
        </function>
        <function type="spring">
          <arg name="bean.name">deleteIndex</arg>
          <arg name="idOriIndexing">indexingServicePublic</arg>
        </function>
      </post-functions>
    </action>
  </actions>
</step>
</steps>
</workflow>

```

Vous noterez que ce fichier est en fait à mettre en relation avec un certain nombre d'autres fichiers de configuration : la configuration des rôles et permissions (le fait qu'un rôle Moderator, Administrator, SuperAdministrator, SuperValidator existe ou pas se configure par exemple), la configuration des

permissions en fonction des formulaires (qui vont permettre pour un rôle donné, dans un état donné, de proposer ou non tel ou tel formulaire), etc.

- le workflow "easy" est utilisé pour le LOM. Un formulaire lom-author-light est configuré pour être utilisable si l'utilisateur a la permission USE_CREATE_FORM sur la fiche considérée. Un formulaire lom-full est configuré pour être utilisable si l'utilisateur a la permission USE_LOM_FORM sur la fiche considérée. (cf le fichier spring-metadata-types.xml)



- Voici ce que donne la version XML (implémentation OsWorkflow donc) proposée par défaut (conf/properties/spring/osworkflow/workflows/workflow_easy.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE workflow PUBLIC "-//OpenSymphony Group//DTD OSWorkflow 2.6//EN" "http://
www.opensymphony.com/osworkflow/workflow_2_8.dtd">
<workflow>
  <initial-actions>
    <action id="100" name="Start Workflow">
      <restrict-to>
        <conditions type="AND">
          <condition type="spring">
            <arg name="bean.name">hasRole</arg>
            <arg name="mask">1</arg><!-- AUTHENTICATED -->
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result old-status="Finished"
          status="Underway" step="1" />
      </results>
      <post-functions>
        <function type="spring">
          <arg name="bean.name">addRole</arg>
          <arg name="mask">2</arg><!-- Role 2 = OWNER -->
          <!-- no recipient -> current user -->
        </function>
        <function type="spring">
          <arg name="bean.name">addPermission</arg>
          <arg name="mask">84</arg><!-- Permission 84 = WRITE + DELETE + USE_CREATE_FORM -->
          <arg name="recipient">2</arg><!-- Role 2 = OWNER -->
        </function>
        <function type="spring">
          <arg name="bean.name">setInstanceIdentity</arg>
          <arg name="xpathMetadataIdentity">/lom:lom/lom:metaMetadata/
lom:identifieur[lom:catalog="UNIT"]/lom:entry</arg>
        </function>
        <function type="spring">
          <arg name="bean.name">setEntryIdentity</arg>
          <arg name="xpathMetadataIdentity">/lom:lom/lom:general/
lom:identifieur[lom:catalog="URI"]/lom:entry</arg>
        </function>
      </post-functions>
    </action>
  </initial-actions>
  <steps>
    <step id="1" name="Private">
      <actions>
```

```

<action id="1" name="Ask to Publish">
  <restrict-to>
    <conditions type="AND">
      <condition type="spring">
        <arg name="bean.name">hasRole</arg>
        <arg name="mask">2</arg><!-- Role 2 = OWNER -->
      </condition>
      <condition type="spring">
        <arg name="bean.name">verifyXPathes</arg>
        <arg name="annotation">
          <span class="warning">Votre Fiche n'est pas complète, vous ne pouvez pas
demander sa publication.</span>
        </arg>
        <arg name="xpath1">/lom:lom/lom:general/lom:title/lom:string[normalize-
space(.) != '']</arg>
        <arg name="xpath5">/lom:lom/lom:technical/lom:location[normalize-space(.) !=
'']</arg>
        <arg name="xpath2">/lom:lom/lom:general/lom:keyword/lom:string[normalize-
space(.) != '']</arg>
        <arg name="xpath4">/lom:lom/lom:lifeCycle/lom:contribute[lom:role/
lom:value='author']/lom:date/lom:dateTime[normalize-space(.) != '']</arg>
        <arg name="xpath3">/lom:lom/lom:lifeCycle/lom:contribute[lom:role/
lom:value='author']/lom:entity[normalize-space(.) != '']</arg>
        <arg name="xpath6">/lom:lom/lom:classification[lom:purpose/
lom:value='discipline']/lom:taxonPath[lom:source/lom:string='Classification UNIT']/lom:taxon/
lom:entry/lom:string[normalize-space(.) != '']</arg>
      </condition>
    </conditions>
  </restrict-to>
  <results>
    <unconditional-result status="Underway"
      old-status="Finished" step="2" />
  </results>
  <post-functions>
    <function type="spring">
      <arg name="bean.name">deletePermission</arg>
      <arg name="mask">84</arg><!-- Permission 84 = WRITE + DELETE + USE_CREATE_FORM
-->
      <arg name="recipient">2</arg><!-- Role 6 = OWNER -->
    </function>
    <function type="spring">
      <arg name="bean.name">addPermission</arg>
      <arg name="mask">148</arg><!-- Permission 148 = WRITE + DELETE + USE_LOM_FORM --
>
      <arg name="recipient">4</arg><!-- Role 4 = MODERATOR -->
    </function>
    <function type="spring">
      <arg name="bean.name">saveXmlHistory</arg>
    </function>
    <function type="spring">
      <arg name="bean.name">sendEmail</arg>
      <arg name="message">
        {0} vient de placer sa fiche en attente de publication
        -----
        Voici ses observations.
        {2}
      </arg>
      <arg name="smtpToMaskRole">4</arg>
      <!-- Role 4 = MODERATOR -->
    </function>
  </post-functions>
</action>
</actions>

```

```

</step>
<step id="2" name="Pending">
  <actions>
    <action id="2" name="Publish">
      <restrict-to>
        <conditions type="AND">
          <condition type="spring">
            <arg name="bean.name">hasRole</arg>
            <arg name="mask">4</arg><!-- Role 4 = MODERATOR -->
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result status="Underway"
          old-status="Finished" step="3" />
      </results>
      <post-functions>
        <function type="spring">
          <arg name="bean.name">deletePermission</arg>
          <arg name="mask">148</arg><!-- Permission 148 = WRITE + DELETE + USE_LOM_FORM --
>
          <arg name="recipient">4</arg><!-- Role 4 = MODERATOR -->
        </function>
        <function type="spring">
          <arg name="bean.name">saveOrUpdateIndex</arg>
          <arg name="idOriIndexing">indexingServicePublic</arg>
        </function>
        <function type="spring">
          <arg name="bean.name">sendEmail</arg>
          <arg name="message">
            {0} vient de publier votre fiche.
            -----
            Voici ses observations.
            {2}
          </arg>
          <arg name="smtpToMaskRole">2</arg>
          <!-- Role 2 = OWNER -->
        </function>
      </post-functions>
    </action>
    <action id="3" name="Reject">
      <restrict-to>
        <conditions type="AND">
          <condition type="spring">
            <arg name="bean.name">hasRole</arg>
            <arg name="mask">6</arg><!-- Role 6 = MODERATOR or OWner : 2 + 4 -->
          </condition>
        </conditions>
      </restrict-to>
      <results>
        <unconditional-result status="Underway"
          old-status="Finished" step="1" />
      </results>
      <post-functions>
        <function type="spring">
          <arg name="bean.name">deletePermission</arg>
          <arg name="mask">148</arg><!-- Permission 148 = WRITE + DELETE + USE_LOM_FORM --
>
          <arg name="recipient">4</arg><!-- Role 4 = MODERATOR -->
        </function>
        <function type="spring">
          <arg name="bean.name">addPermission</arg>
          <arg name="mask">84</arg><!-- Permission 84 = WRITE + DELETE + USE_CREATE_FORM
-->

```

```

        <arg name="recipient">2</arg><!-- Role 2 = OWNER -->
    </function>
    <function type="spring">
        <arg name="bean.name">sendEmail</arg>
        <arg name="message">

                {0} vient de refuser une fiche.
                -----
                Voici ses observations.
                {2}

        </arg>
        <arg name="smtpToMaskRole">6</arg>
        <!-- Role 6 = MODERATOR or OWner : 2 + 4 -->
    </function>
    </post-functions>
</action>
</actions>
</step>
<step id="3" name="Public">
    <actions>
        <action id="4" name="Make Private">
            <restrict-to>
                <conditions type="AND">
                    <condition type="spring">
                        <arg name="bean.name">hasRole</arg>
                        <arg name="mask">2</arg><!-- Role 2 = OWNER -->
                    </condition>
                </conditions>
            </restrict-to>
            <results>
                <unconditional-result old-status="Finished"
                    status="Underway" step="1" />
            </results>
            <post-functions>
                <function type="spring">
                    <arg name="bean.name">addPermission</arg>
                    <arg name="mask">84</arg><!-- Permission 84 = WRITE + DELETE + USE_CREATE_FORM
-->

                <arg name="recipient">2</arg><!-- Role 2 = OWNER -->
            </function>
            <function type="spring">
                <arg name="bean.name">deleteIndex</arg>
                <arg name="idOriIndexing">indexingServicePublic</arg>
            </function>
            </post-functions>
        </action>
    </actions>
</step>
</steps>

</workflow>

```

On notera que l'on a placé dans la distribution des fichiers exemple-* qui proposent des configurations plus complexes à titre d'exemple (dont la mise en place du workflow présenté plus haut : workflow "complex" (anciennement appelé "model_1"))

Installation / migration / maintenance via ANT

Une fois les différents fichiers de configuration modifiés pour correspondre à vos besoins, vous pouvez enfin installer/déployer/initialiser votre application dans le tomcat cible (cf build.properties). Pour cela un certain nombre de targets ant est à votre disposition : * clean: pour supprimer les fichiers compilés et déployés.

- deploy: permet de compiler et déployer l'application dans le tomcat.
- init: pour initialiser la base de données sql : notez que la base doit déjà exister sur le serveur sql et que cela ne crée pas de collections sur la base eXist (ATTENTION: si des tables pré-existent elles sont écrasées !).
- update-acls: permet d'informer la base de données des modifications des permissions/rôles faites dans les fichiers de configuration.
- upgrade: mets à jour la base de données entre 2 versions d'ori-oai-workflow-spring
- importmetadatas: pour importer des métadonnées depuis un zip contenant des fichiers xml de métadonnées (expérimental dans cette version ori-oai-workflow-spring-)
- local-reindexall: permet de mettre à jour l'index local de l'application ori-oai-workflow. ori-oai-workflow utilise en effet un index local (Compass/Lucène) pour permettre une navigation et une recherche rapide dans les fiches. Cet index est indispensable au bon fonctionnement de l'application. Il est synchronisé avec la base de données. Dans certains cas il peut être utile de le reconstruire complètement (resynchroniser) avec la base de données. C'est le but de cette target. On fera un ant clear-index avant. On ne peut faire cette opération que lorsque le module est arrêté.
- clear-index: confère la target local-reindexall, la target clear-index détruit l'index local.
- reindexall: demande au module ori-oai-indexing de réindexer toutes les fiches qui sont censées être indexées, c'est à dire qui sont marquées comme étant indexées. Cela peut être utile lors de la reconstruction complète de l'index du module ori-oai-indexing.

Pour une installation, vous devrez normalement appeler les targets suivantes dans l'ordre : *
deploy,

- init,
- update-acls.

Avant cela vous aurez : * configuré les différents fichiers,

- démarré la base de données sql (en y créant la base de données donnée dans le fichier de conf du module),

• ...

Après l'appel des targets, vous pourrez démarrer le serveur tomcat. Pour une mise à jour (attention de ne pas appeler la target init qui initialise votre base de données, vous devrez normalement appeler les targets suivantes dans l'ordre (l'application ori-oai-workflow le tomcat doit alors être arrêté) : * clean,

- upgrade,
- deploy.

Au préalable vous aurez * fait des sauvegardes de la base MySQL (et de la base eXist également si vous êtes en version 1.0.*),

- fait une sauvegarde de vos sources et configurations du module
- fait un switch subversion pour passer du tag précédent à ce nouveau tag au niveau de vos sources (si vous utilisez subversion pour synchroniser vos sources au niveau de l'intégration/exploitation ORI-OAI),



Utiliser les possibilités de svn switch pour passer d'un tag à un autre si vous êtes branché via Subversion. Cela vous permet de fusionner vos modifications (fichiers de config et autres) et les modifications apportées dans la nouvelle version astucieusement. Subversion directement en ligne de commandes est le moyen le plus confortable. => le fichier CHANGELOG est à consulter à chaque mise à jour - il tente de vous apporter un certain nombre d'informations sur les changements opérés et que vous devrez donc prendre en compte.

- résolu les conflits et vérifié (en vérifiant rapidement vos configurations/personnalisations propres)
- vérifié que les bases (SQL et éventuellement eXist si vous êtes en version 1.0.*) tournent



Les librairies pluto-1.0.1-rc4.jar et portlet-api-1.0.jar qui servent en fait pour l'utilisation de ori-oai-workflow-spring en mode portlet doivent être ajoutées manuellement à vos librairies Tomcat si vous souhaitez faire tourner le module dans un container de portlet. Il faudra également décommenter les déclarations relatives à la servlet "portlet" dans le web.xml commentées par défaut.

Si vous souhaitez utiliser CAS, il faut comme indiqué dans les commentaires de main-config.properties configurer le web.xml sauf si vous passez par la facilité du Quick\Install qui fait alors cela pour vous. Si vous utilisez un certificat non reconnu par la JVM (auto-signé pour des tests par exemple), il faut également ne pas oublier d'ajouter un trustStore et de le spécifier dans les paramètres Tomcat lors du lancement de celui-ci :

```
CATALINA_OPTS="-Djavax.net.ssl.trustStore=E:/Java/ac-racine-cru.keystore"
```

=> L'utilisation d'un tel certificat est bien évidemment à déconseiller.

Shibboleth

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Authentification / Autorisation avec Shibboleth Nouveau

Depuis la version 1.?.? ori-oai-workflow peut utiliser Shibboleth à la fois pour l'authentification et l'autorisation. LDAP et CAS ne sont alors plus nécessaires et il est possible de partager une même instance d'ori-oai-workflow entre plusieurs établissements.

Pour que les mécanismes shibboleth soient pris en compte au niveau de ori-oai-workflow, ce dernier attend de pouvoir interpréter directement les attributs Shibboleth positionnés dans les paramètres HTTP de la requête cliente. On peut pour ce faire utiliser directement le mod_shib d'apache, dans lequel on requerra une authentification Shibboleth sur l'ensemble de l'applicatif.

Voici un exemple d'une telle configuration (le mod_shib et proxy_ajp (ProxyPass) sont ici utilisés et donc chargés par ailleurs :

```
ProxyPass /ori-oai-workflow-spring ajp://localhost:8009/ori-oai-workflow-spring

<Location /ori-oai-workflow-spring>
  AuthType shibboleth
  ShibRequireSession On
  ShibUseHeaders On
  require valid-user
</Location>
debian:/etc/apache2# more conf.d/workflow.conf
ProxyPass /ori-oai-workflow-spring ajp://localhost:8009/ori-oai-workflow-spring

<Location /ori-oai-workflow-spring>
  AuthType shibboleth
  ShibRequireSession On
  ShibUseHeaders On
  require valid-user
</Location>
```

Reste ensuite à configurer l'interprétation des attributs Shibboleth au niveau ori-oai-workflow : on va pouvoir définir d'une part l'attribut Shibboleth qui va correspondre à l'identifiant d'un individu dans ori-oai-workflow, puis d'autres part les règles qui vont permettre de définir à quel(s) groupe(s) appartient l'individu connecté; ces règles, à paramétrer par l'exploitant, devront s'appuyer sur les attributs shibboleth disponibles.

Ces 2 configurations se font dans le fichier conf/properties/spring/acegi/acegi-authentication-shib.xml. Comme décrit rapidement dans le fichier, les règles doivent correspondre à des expressions Java renvoyant un boolean (true, false). Si l'expression interprétée renvoie true, alors on affecte à l'individu connecté le groupe donné en clé. shibAttrsMap est le nom de la variable définissant une table de hashage regroupant l'ensemble des attributs Shibboleth disponibles : nom/valeur.

```
<bean name="shibAuthenticator"
class="org.orioai.workflow.services.application.shib.ShibAuthenticator">
  <property name="remoteUserShibAttr" value="Shib-InetOrgPerson-mail"/>
  <property name="shibGroupsRules">
    <description>
      shibGroupsRules defines a Map where
        * keys are groups names
        * values are java expressions (evaluated by Janino) which
should return true or false -
      to construct expressions, you have a parameter named
shibAttrsMap. Tis Map contains key/value where :
        * keys are the shib parameter name
        * values are the shib parameter value
```

```

        </description>
        <map>
            <entry key="all">
                <value>true</value>
            </entry>
            <entry key="moderators">
                <value>shibAttrsMap.get("Shib-EP-
UnscopedAffiliation").equals("faculty;member") && shibAttrsMap.get("Shib-EP-
PrimaryAffiliation").equals("professor")</value>
            </entry>
            <entry key="admins">
                <value>shibAttrsMap.get("Shib-InetOrgPerson-
mail").equals("Vincent.Bonamy@univ-rennes1.fr")</value>
            </entry>
        </map>
    </property>
</bean>

```

Limitations: de par le mode de fonctionnement de shibboleth, il n'est pas possible de retrouver dynamiquement tous les individus d'un groupe donné. On ne pourra donc pas utiliser dans un workflow l'envoi de mails à l'ensemble des individus d'un groupe. On contournera ce problème en définissant directement les individus un à un comme destinataires des mails, ou mieux encore en définissant en amont une mailing-list permettant cet envoi de mails à un ensemble prédéfini d'individus.

Utilisation

This page last changed on Jan 05, 2009 by ycolmant@univ-valenciennes.fr.

Utilisation - Référencer un document

- [Captures d'écran](#)

Note : Nous présentons dans ce document une utilisation simple du module ORI-OAI-workflow. Vous pouvez également visualiser d'autres captures d'écran à l'adresse <http://sourcesup.cru.fr/ori-workflow/1.1/utilisation.html>.

Avec votre navigateur, vous pouvez vous connecter à :

```
http://[HOST_INSTALL]:8185/ori-oai-workflow
```

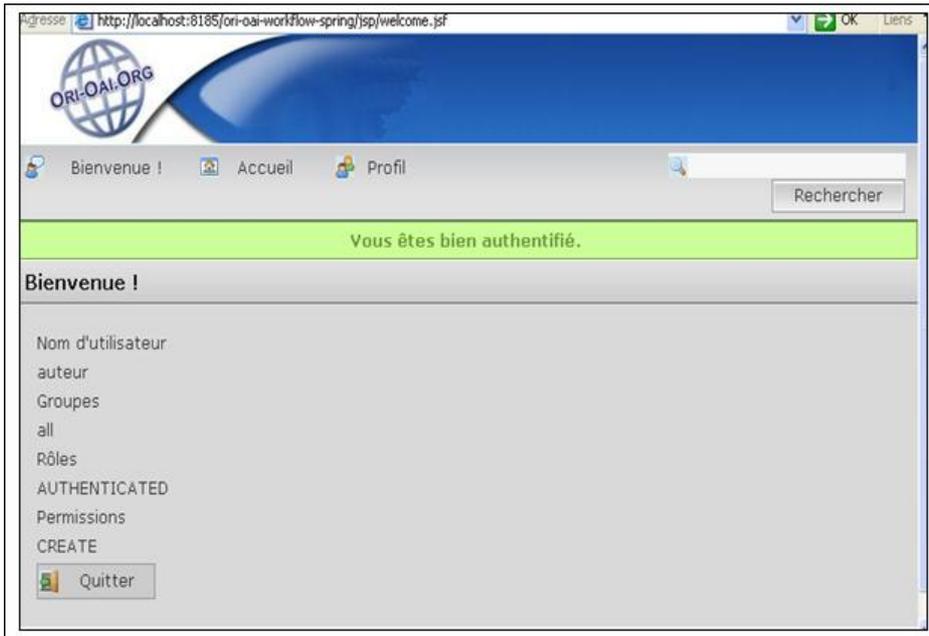
et tester l'application.

Le workflow proposé par défaut pour les fiches LOM comporte trois états :

- privé,
- en attente de publication,
- publié.



Identifiez-vous avec un login ayant le rôle **auteur**. (**login_auteur**)



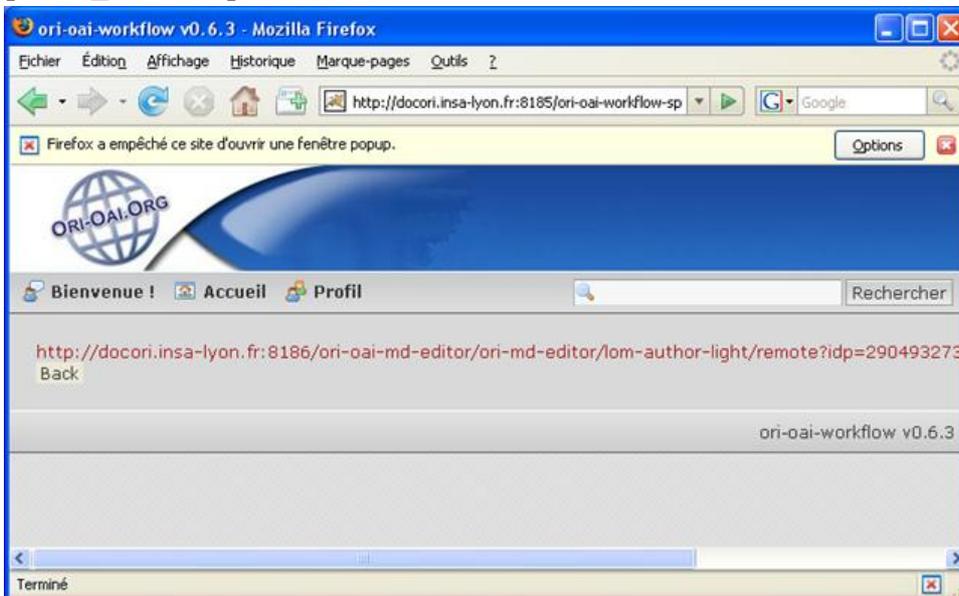
Cliquez sur « Accueil ».

Dans la partie « Référencer une Ressource », choisissez le type de métadonnées que vous souhaitez créer (par exemple « **Ressource Pédagogique [LOM]** »).

Vous aurez peut-être une alerte indiquant que le navigateur a refusé d'ouvrir une fenêtre.

Sous Firefox : **Firefox a empêché ce site d'ouvrir une fenêtre popup**

Cliquez sur **Options** (dans le petit bandeau jaune), puis sur **Autoriser les popup pour [HOST_INSTALL]**



Remplissez le formulaire, lorsque celui ci **est rempli et validé** (le message d'erreur en haut du formulaire a disparu), cliquez sur « **Sauvegarder** ».

Les champs obligatoires sont en rose.

http://docori.insa-lyon.fr:8186 - ORI-MD-Editor - Mozilla Firefox

Editeur ORI-MD

Français

Quitter Sauvegarder Sauvegarder et Quitter

Localisation ? bibliothèque

Ajout d'une localisation

Titre ? le langage PHP

Mots-clés libres ? programmation

Ajout d'un mot-clé

Classification UNIT ?

Id 13 ?

Entrée Informatique ?

Recherche de Taxonomie

Ajout d'un taxon

Auteur ?

Prénom Recherche de vCard

Nom Doc'INSA ?

Email ?

Organisation ?

Date de création

Wednesday June 27, 2007 ?

2007-06-27 < J-JM-AAAA

Editeur ?

Prénom Recherche de vCard

Nom

Terminé

Vous pourrez revenir sur ce formulaire ultérieurement si vous le souhaitez.

Pour Demander la publication des métadonnées que vous avez saisies, cliquez sur **Mes ressources en cours d'édition [1]**.

Adresse http://localhost:8185/ori-oai-workflow-spring/jsp/listing-instances.jsf

OK Liens

ORI-OAI.ORG

Bienvenue ! Accueil Profil

Rechercher

Vous êtes ici: Accueil

Liste des ressources en cours de traitement ou traitées

Mes ressources en cours d'édition [1]

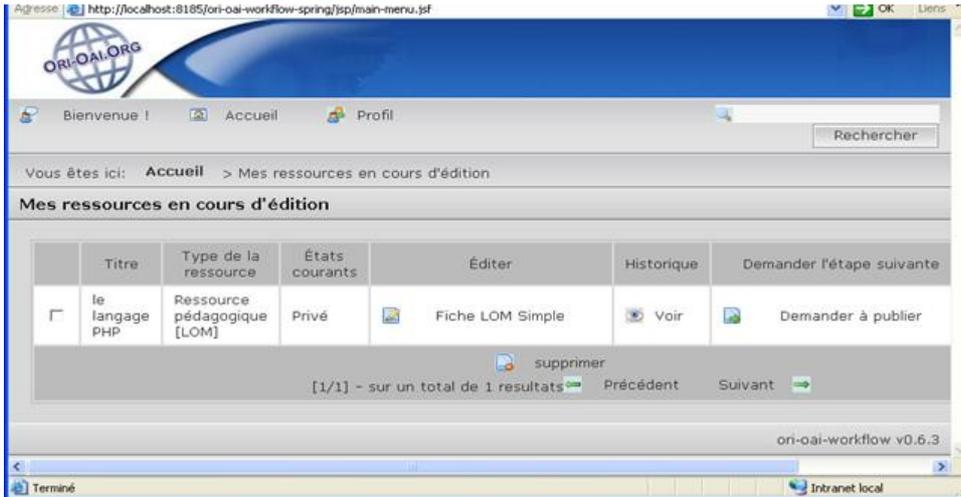
Référencer une ressource

Ressource pédagogique [LOM]

Document [DC]

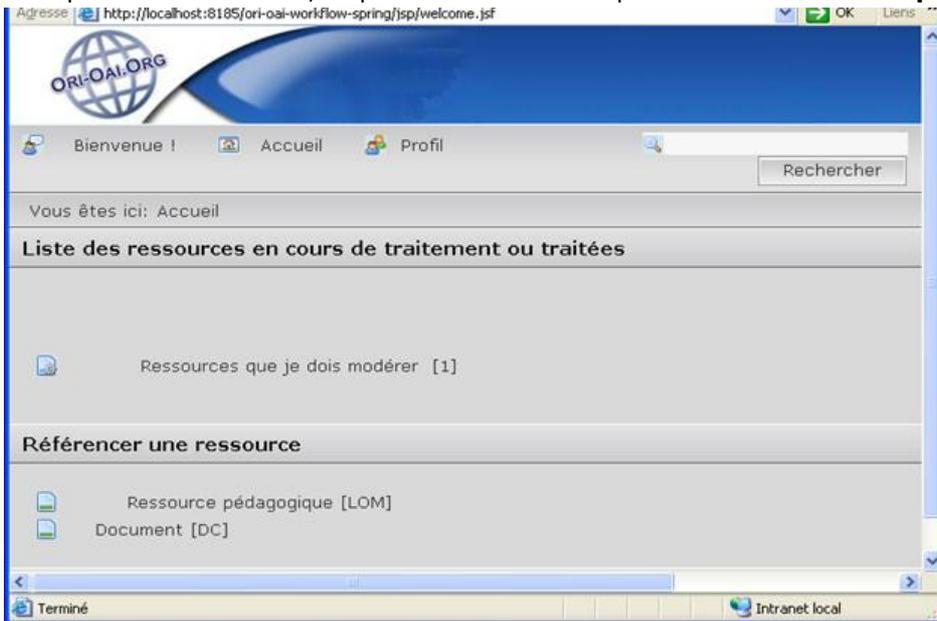
Terminé Intranet local

Le document est dans l'état « **privé** ».



Pour passer à l'état « **en attente de publication** », cliquez sur « **demander à publier** ». Vous pouvez rédiger une observation, puis cliquez sur « **ok** ».
Déconnectez-vous de l'utilisateur ayant le rôle Auteur (clic sur Bienvenue, puis Quitter) et connectez-vous avec un utilisateur ayant le rôle **Modérateur (login_moderateur)**.

Pour publier le document, cliquez sur « **Accueil** » puis sur « **Ressources que je dois modérer [1]** ».



Cliquez sur « **Publier** », vous pouvez rédiger une observation, puis cliquez sur « **ok** ».



Vos données sont maintenant publiées.

Captures d'écran

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Error trying to draw image gallery org.apache.velocity.exception.MethodInvocationException: Invocation of method 'requireResource' in class \$Proxy40 threw exception java.lang.NullPointerException @ /com/atlassian/confluence/plugins/macros/advanced/gallery-default.vm[1,21]

Aspects pratiques

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

- [Tests ANT](#)
- [Eclipse](#)
- [Encodage](#)
- [Sauvegardes](#)
- [Sécurisation des Web Services](#)

Tests ANT

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Tests ANT

Test de l'environnement et des configurations ...

Avant de déployer votre application, vous pouvez lancer un test d'appel aux principaux services utilisés et censés être configurés dans votre Workflow. Ceci permet d'identifier une misconfiguration, un service non lancé (ldap, eXist, ...) ...

Placez-vous dans le répertoire « **[ORI_HOME]/src/ori-oai-workflow-svn** ».

```
ant remoteconfigtest
```

Si tout fonctionne bien, voici ce que vous devriez obtenir à peu de chose près :

```
remoteconfigtest:
[junit] Testsuite: org.orioai.workflow.RemoteConfigTest
[junit] Tests run: 5, Failures: 0, Errors: 0, Time elapsed: 11,761 sec
[junit] ----- Standard Output -----
[junit] Retrieving document at 'null'.
[junit] Retrieving document at 'null'.
[junit] -----
[junit]
[junit] Testcase: testGetVocabulariesId took 10,638 sec
[junit] Testcase: testSearchIndexingServices took 0,986 sec
[junit] Testcase: testInitialDirContextFactory took 0,049 sec
[junit] Testcase: testConnectUrls took 0,048 sec
[junit] Testcase: testSmtpt took 0,033 sec

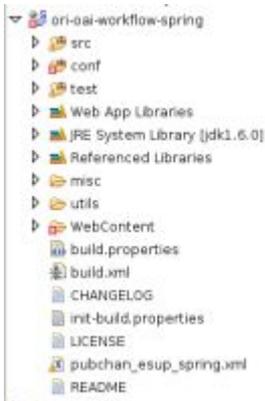
BUILD SUCCESSFUL
```

Eclipse

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

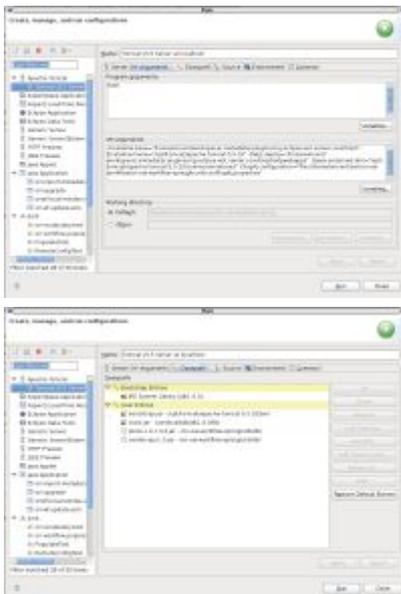
ori-oai-workflow-spring: environnement de développement - Eclipse

Partie adressée plutôt aux développeurs Le développement dans ori-oai-workflow-spring peut s'appuyer sur eclipse. La structuration des sources est structurée dans ce sens. Les fichiers .classpath .project .springBeans sont donnés dans les répertoires subversion (tags et trunk) pour une intégration facile dans eclipse. Pour ce faire, l'idée est de récupérer le projet (tag ou trunk), de l'intégrer en tant que nouveau "projet Web". Ci-dessous une copie d'écran montrant ce que l'on doit alors obtenir (le buid path est normalement configuré de suite comme il se doit, etc. cela grâce aux fichiers .classpath et .project.



On s'appuie sur WTP pour lancer le module depuis le eclipse. Pour cela, on configure l'exécution du projet via la vue serveur en le paramétrant pour l'exécuter dans un serveur Tomcat (5 ou 6 au choix).

Les configurations spécifiques à l'exécution de cela dont données via ces copies d'écran.



Encodage

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Encodage

De par leurs fonctionnalités et leurs objectifs d'interopérabilité, les modules ORI-OAI sont voués à fonctionner dans un encodage UTF-8. Il est préférable (pour éviter au mieux les problèmes d'encodage dans les données, le rendu etc.) de positionner l'encodage à UTF-8 pour tous les composants acteurs dans ORI-OAI (les bases de données, les serveurs d'application, etc). Cela peut se faire de différentes manières (depuis les variables d'environnement du système par exemple LANG). Le positionner en tant qu'option Tomcat est une bonne option également :

```
CATALINA_OPTS=-Dfile.encoding=UTF-8
```

Sauvegardes

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Sauvegardes

Ori-Oai-Workflow utilise 1 base de données SQL (cf les pré-requis de l'installation/configuration d'Ori-Oai-Workflow Spring). Il faut donc penser à sauvegarder cette base de données très régulièrement, pour palier les potentiels pannes système, pbs disque dur, ...

MySql

Vous trouverez plus d'infos ici simplement : [CommentCaMarche : Importer et exporter des données sous MySQL](#) par exemple (sinon [la doc officielle](#)).

```
mysqldump --all-databases -u root > /opt/ori/backup_mysql.dump
```

Autres ...

Si votre choix s'est porté sur une autre Base supporté par Hibernate (PostgreSql par exemple), la doc officielle de cette Base de Données devrait également vous permettre de mettre en oeuvre les sauvegardes adéquates.

Sécurisation des Web Services

This page last changed on Oct 14, 2008 by vibonamy@univ-rennes1.fr.

Sécurisation des Web Services

De même que pour l'encodage, la sécurisation des Web Services est un problème commun à tous les modules. Il peut être opportun de sécuriser les Web Services qui n'ont pas vocation à être utilisés par le monde entier (il faut y penser). Notamment les Web Services du Workflow, de l'indexing, etc. Certaines pages comme la page permettant le rechargement des vocabulaires dans ori-oai-md-editor doivent être sécurisées également.

Pour ce faire, le plus simple et le plus efficace est de sécuriser ces urls à un niveau assez "bas", un niveau "système" : * au niveau de la configuration du Tomcat voir de Apache par exemple (username/password en authentication basic http) côté serveur et au niveau du web.xml de l'application (dans les configs de XFire notamment) côté client.

- en n'ouvrant pas le port utilisé par le serveur d'application de l'indexing par exemple.
- en protégeant les accès via des contrôles sur les IPs
- etc.