OSUG-DOI (public)

http://doi.osug.fr/

Principe

Roles:

- Datacite: assure la résolution du DOI et possède les informations pour chaque DOI: méta données (xml) + URL associée vers doi.osug.fr (landing page ou ressource)
- OSUG-DC: le serveur doi.osug.fr héberge les landing pages (en local) ou réalise une redirection vers un autre serveur (spip...) dans le périmètre de l'OSUG
- les landing pages doivent être pérennes et contenir les instructions pour accéder aux données / services

Développement

Source code: https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi

TODO:

- DOC:
  1. écrire la Procédure pour demander des DOIs (intro, approche, méta-données), initier un projet (templates, config...)
  2. créer template CSV (with all possible key/values + comments)
- DEV:
  1. supervision pour detecter les URLs invalides (URL 404) et basculer sur les landing page locales
- PROD:
  1. recréer vm doi.osug.fr (ansible) et déploiement application automatisé

Architecture:

Génération Landing Pages (datacite XML to HTML)

A partir d'un document XML correspondant aux métadonnées du DOI (datamodel datacite), une transformation XSLT génère une page HTML statique (bootstrap + CSS) qui présente simplement les méta données et les instructions d'accès aux données / service.

Outil de génération des métadonnées DOI (datacite XML) à partir des méta

données ISO-19139 (CSW)

Cas d'utilisation = AMMA-CATCH (voir http://bd.amma-catch.org/) ou OHMCV (fiches Sedoo)

Principe

Certaines infos nécessaires au schéma datacite sont déjà renseignées pour chaque jeu sur le portail Web d'accès aux données AMMA-CATCH et sont récupérables en utilisant le webservice CSW méthode getRecords (format XML ISO 19139):
http://bd.amma-catch.org/amma-catchWS2/WS/csw/default?service=CSW&request=GetRecords&version=2.0.2&typenames=csw:Record&resulttype=results&maxrecords=100&elementsetname=summary&outputschema=http://www.isotc211.org/2005/gmd

Il faut ensuite les compléter pour produire un fichier XML complet (dataCite) pour chaque jeu de données.

Chaine de fusion des informations (CSW + templates + URLs)

Toute la configuration du service OSUG-DOI est archivée dans git (forge gricad):
https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi/tree/master/conf

Pour compléter les metadonnées obtenues du webservice CSW, des templates (global, par pays, par jeu de données) au format CSV sont utilisés:
https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi/tree/master/conf/AMMA-CATCH/templates

Pour indiquer l'URL d'une landing page externe (spip...), le fichier doi_url_landing_page.csv (table [DOI suffix :: URL (absolue)]) est utilisé:
https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi/blob/master/conf/AMMA-CATCH/doi_url_landing_page.csv

Pour donner l'URL associée réellement au DOI (accès aux donnés), le fichier doi_url_data_access.csv (table [DOI suffix :: URL (absolue)]) est utilisé:
https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi/blob/master/conf/OHMCV/doi_url_data_access.csv Si le DOI ou ce fichier est absent, la valeur par défaut définie dans le fichier project.properties (dataAccessUrl=...) est utilisée.

Pour définir les instructions d'accès aux données, il faut fournir le fragment HTML:
https://gricad-gitlab.univ-grenoble-alpes.fr/OSUG/DC/osug-doi/blob/master/conf/AMMA-CATCH/access_instruction.html

Séquence de traitement:

1. GetRecords → renvoie autant d'élements <gmd:MD_Metadata></gmd:MD_Metadata> qu'il y a de jeux de données

2. Conversion au format CSV
3. Pour chaque jeu de données, fusionner avec les templates (ajout, pas de suppression):
    1. template global: template_all.csv
    2. template du pays: correspondant au pays indiqué par geoLocationPlace (geoLocationPlace;Benin)
    3. template spécifique au jeu de données: correspondant au suffixe du DOI (identifier:DOI;10.5072/AMMA-CATCH.CL.Run_O)
    4. sauvegarder et convertir au format XML datacite
4. validation des fichiers XML avec le schéma datacite
5. vérifier les related identifiers (existe en BDD ?) pour tester des références invalides ?
6. vérifier les URLS (landing page externe et l'URL associée au DOI)
7. générer les landing pages associées (html) avec l'accès aux données (fragment html + URL)
8. tester publication du DOI (metadata + URL OSUG-DOI) avec le préfixe de test: STAGING
9. sauvegarder les informations du DOI en BDD (statut, validation, URL ...)

Notes:

1. les identifiants DOI présents dans la configuration (CSV) ne doivent contenir que le préfixe de test (publication ultérieure)
2. les méta-données en doublons sont ignorées (même clé / valeur) mais attention si les informations ne sont pas disjointes (clés présentes dans plusieurs templates)
3. attention à l'encodage des fichiers CSV (UTF-8 sous linux et iso-8859-1 sous windows) !
4. il est possible d'utiliser des fichiers Excel pour saisir les templates (qui seront convertis correctement en CSV)

Chaine de fusion des informations (CSW + templates)

Gestion des projets

Gestion multi projets sur le serveur:

- conf (origine: git)
    - AMMA-CATCH
    - OHMCV
    - projet ...
- staging
    - AMMA-CATCH
    - OHMCV
    - projetX
- public
    - AMMA-CATCH
    - OHMCV

Publication DOI (datacite)

objectif: automatiser la publication des DOIs et alimenter la BDD (statut)

TODO:

1. copier la landing page (URL publiée chez datacite): PUBLIC
2. reécrire les identifiants DOI (identifier et relatedIdentifier) pour remplacer le préfixe de test en préfixe OSUG à l'aide de la base de données
3. publier ce DOI avec le préfixe OSUG et mettre à jour la BDD
4. gérer les redirections ie une URL alternative pour la landing page

note: garder les documents XML + URL dans le file system et BDD