# OWL2YAMS : créer une application CubicWeb à partir d'une ontologie OWL

Fabien Amarger, Nicolas Chauvat, Elodie Thiéblin

Logilab, 104 boulevard Louis-Auguste Blanqui 75013, Paris

prénom.nom@logilab.fr

## Résumé

*CubicWeb est un cadriciel Web qui facilite le développement de systèmes de gestion de contenus sémantiques (SCMS), lesquels permettent la publication de données par négociation de contenu HTTP et leur administration avec les fonctions attendues d'un CMS classique. Une application CubicWeb est basée sur un schéma YAMS qui définit la modélisation métier des données gérées. OWL est le standard du W3C pour décrire des modèles de données. Nous présentons dans cet article l'outil OWL2YAMS qui permet de créer une application CubicWeb à partir d'une ontologie OWL, puis de charger dans cette application des données décrites en RDF avec les termes de cette ontologie.*

## Mots-clés

*CubicWeb, Web de données liées, négociation de contenu, publication RDF*

## Abstract

*CubicWeb is a semantic framework dealing with data publication, content negociation and data administration, as well as providing Web framework basic functionalities. A CubicWeb application is based on a YAMS schema to model the data. OWL is the W3C standard for representing data model. In this article, we present OWL2YAMS to create a CubicWeb application from an OWL ontology. This tool also enables the loading of data encoded in RDF with the vocabulary of this ontology.*

## Keywords

*CubicWeb, Linked data, content negociation, RDF publishing*

# 1 Introduction

Les données du Web de données liées sont très souvent publiées dans un entrepôt SPARQL ou en utilisant un fichier "dump" contenant tous les triplets RDF. Il est plus rare de pouvoir accéder aux données à travers la négociation de contenu en déréférençant les URI. De plus, nous ne connaissons pas d'interface utilisateur permettant de manipuler des données RDF à la manière d'un CMS (opérations CRUD [1], définitions fines des permissions, rendu graphique, etc.). C'est pour cela que nous présentons CubicWeb, notre cadriciel pour développer des systèmes de gestion de contenu (CMS [2]) sémantiques, accompagné de OWL2YAMS qui permet de créer une application CubicWeb à partir d'une ontologie OWL et d'importer des données en RDF. Il devient ainsi très simple de publier des données RDF et une ontologie OWL pour les intégrer au Web des données liées, en utilisant la négociation de contenu et en bénéficiant de toutes les fonctionnalités et interfaces d'administration que l'on peut attendre d'un CMS sémantique.

# 2 CubicWeb

CubicWeb est un logiciel libre écrit en Python, dont le développement a commencé en 2001, l'année de publication de l'article fondateur du Web Sémantique [1]. CubicWeb a été conçu pour faciliter le développement et le déploiement d'applications qui reprennent les concepts essentiels du Web Sémantique. Avec CubicWeb il est aisé de gérer et de rendre accessibles des données qui suivent un modèle préalablement défini.

CubicWeb utilise le formalisme YAMS (Yet Another Magic Schema [3] pour représenter le modèle de données de façon explicite. Observons sur la figure 1 que ce schéma YAMS est utilisé pour générer un modèle de données SQL. De cette façon, le schéma explicite de YAMS s'appuie sur la performance et la stabilité du SQL pour son fonctionnement technique.

Le langage RQL (Relation Query Language [4] est utilisé pour exprimer des requêtes en utilisant les noms des classes et des relations du modèle exprimé en YAMS. La requête RQL est compilée en une requête SQL et exécutée sur la base de données relationnelle. Cette approche permet de développer une application Web en ne manipulant que les termes d'un modèle de données explicite et donc de s'abstraire des contraintes techniques du SQL sous-jacentes. L'avantage est que le modèle YAMS et les requêtes RQL peuvent être présentés aux experts métiers et servir de base

1. Create, Retrieve, Update, Delete

2. https://en.wikipedia.org/wiki/Content_management_system

3. https://forge.extranet.logilab.fr/open-source/yams

4. https://forge.extranet.logilab.fr/cubicweb/RQL

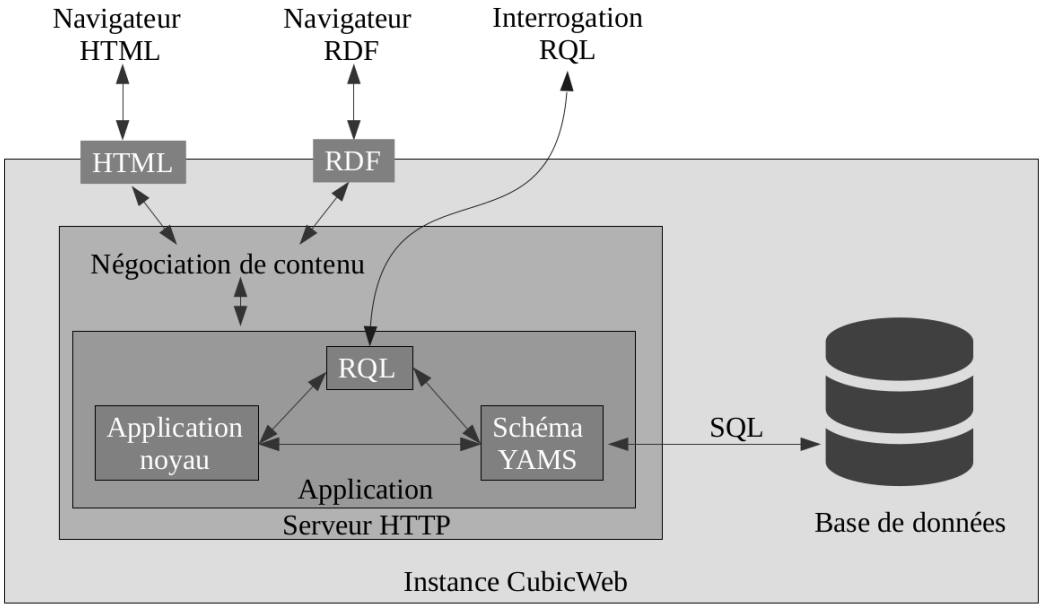


FIGURE 1 – Schéma d'une instance CubicWeb en fonctionnement

de discussion commune, alors que les contraintes techniques du SQL (normalisation, tables de jointure, clef étrangère, etc.) obscurciraient le tableau et rendraient la discussion plus difficile.

Le formalisme YAMS permet la définition de contraintes fortes sur les données (cardinalités, permissions en lecture et/ou en écriture, etc.). Un grand nombre de contraintes métiers peuvent donc être définies directement dans le modèle de données avec la garantie d'être respectée par les requêtes RQL en lecture et en écriture. L'étape de compilation de RQL en SQL combine en effet le contenu de la requête RQL et les contraintes exprimées dans le modèle YAMS. La même requête RQL exécutée par des utilisateurs ayant des permissions différentes ne produit pas toujours le même SQL et ne retourne pas nécessairement le même résultat.

Une fonctionnalité particulièrement appréciable dans CubicWeb est la possibilité d'utiliser une interface graphique automatiquement générée. Cette interface permet d'administrer les données facilement, de gérer les utilisateurs et utilisatrices, d'afficher les données, etc. Cette interface n'est pas obligatoire et il est tout à fait possible de développer sa propre interface grâce à une API permettant de manipuler les données facilement.

Chaque application développée avec CubicWeb prend la forme d'un *cube*, c'est-à-dire d'un composant applicatif réutilisable définissant un modèle, une logique métier et des règles d'affichage. Pour développer une application, il est possible d'inclure d'autres cubes, ce qui signifie qu'avec de l'organisation des applications peuvent être construites par assemblage de composants fonctionnels élémentaires.

Fort de ces fonctionnalités, CubicWeb a pu répondre aux besoins de projets conséquents tels que DataBnf (https ://-data.bnf.fr) [2], FranceArchives (https ://francearchives.fr/) ou DataPOC (https ://datapoc.mnhn.fr/).

Pour une description plus détaillée de CubicWeb, on pourra lire [3].

# 3 État de l'art

## 3.1 Application de publication sur le Web de données liées

La question principale sur laquelle nous nous penchons est : "comment construire facilement une application basée sur des données en RDF ?" Le premier pas est la publication des données sur le Web [4]. Nous faisons ici l'état de l'art des systèmes permettant la gestion de ces deux étapes : exposer des ressources RDF sur le web et proposer une interface d'administration pour consulter et/ou gérer (créer, modifier ou supprimer) ces ressources.

Dans cet état de l'art, nous avons considéré les propriétés suivantes comme nécessaires à la gestion des données :

**négo. cont.** Si l'outil propose de la négociation de contenu sur les données gérées et quelle forme prend cette négociation de contenu (orientée serveur, agent ou transparente, c'est-à-dire par redirection HTTP).

**SPARQL** Si l'outil propose une API SPARQL pour interroger les données

**RDF** S'il est possible d'intégrer des données existantes en RDF dans l'outil

**perms** Si l'outil propose une gestion des permissions sur les données

**admin.** Si une interface d'administration des données est incluse dans l'outil

**visu.** Si une interface de visualisation des données (pages html) est incluse

**licence** La licence de publication de l'outil

**publication** La date dernière version publiée

| Outil | négo. cont. | SPARQL | RDF | perms. | admin. | visu. | licence | publication | inst. |
|---|---|---|---|---|---|---|---|---|---|
| Apache Marmotta [1] | serveur | ✓ | ✓pour représenter ressources LDP | ✓ | ✓ | ✗ | Apache 2 | 12/06/2018 | ✓ |
| CarbonLDP [2] | serveur | ✓ | ✗ | ✓ | ✓ | ✗ | Propriétaire | 4/10/2018 | ✓Version Gratuite |
| OntoWiki [3] | ✗ | ✓virtuoso dédié | ✓ | ✗ | ✓ | ✓ | GPL | 4/10/2016 | ✗Erreur PHP à l'exécution |
| Open Semantic Framework [4] | ? | ✓ | ? | ✓ | ✓ | ✓ | Apache 2 | 26/02/2015 | ✗Limité à CentOS 6 et 7, Ubuntu 14.04 |
| PSPS [5] | serveur | ✓ | ✓ | ✗ | ✗ | ✓ | MIT License | 17/11/2019 | ✓ |
| Virtuoso [6] | transparent [9] | ✓ | ✓ | ✓ | ✗ | ✓ | GPL | 22/06/2021 | ✓Version OpenSource |
| LinkedDataHub [7] | server (uniquement sur les graphes) | ✓ | ✓ | ✓ | ✓ | ✓ | Apache-2.0 License | 18/02/2022 | ✓ |
| **CubicWeb** [8] | serveur | ✗ | ✓ | ✓ | ✓ | ✓ | LGPL | 10/03/2022 | ✓ |

[1] https://marmotta.apache.org
[2] https://carbonldp.com/
[3] https://docs.ontowiki.net
[4] http://opensemanticframework.org
[5] https://github.com/factsmission/psps
[6] https://virtuoso.openlinksw.com
[7] https://atomgraph.github.io/LinkedDataHub/
[8] https://cubicweb.readthedocs.io/
[9] https://datatracker.ietf.org/doc/html/rfc2295#section-4.3

TABLE 1 – Comparaison d'outils de publication et gestion de données RDF sur le Web

**inst.** Si nous avons réussi à installer l'outil

Le tableau 1 présente le résultat de cette étude.

D'après les contraintes que nous nous étions fixées, nous pouvons observer que CubicWeb est le seul cadriciel proposant de la négociation de contenu, ainsi que la gestion des permissions et une interface de visualisation et d'administration des données. De plus, CubicWeb est aussi toujours maintenu à l'heure actuelle et publié sous une licence libre LGPL. Nous pouvons remarquer néanmoins qu'il ne permet pas d'interroger les données en SPARQL. Ce manque est en cours d'étude, mais comporte quelques problèmes techniques, principalement liés au fait que RQL est bien moins expressif que SPARQL.

Le projet *LinkedDataHub* semble aussi correspondre à nos critères et semble lui aussi maintenu, mais nous avons remarqué quelques limitations. Tout d'abord, la négociation de contenu ne peut se faire que sur un graphe et non pas sur une ressource particulière. De plus, la spécification de la visualisation d'une ressource ne peut se faire qu'à partir de règles CSS [5], ce qui ne permet pas de répondre à tous les besoins en visualisation, notamment lorsqu'une interaction avec l'utilisateur ou l'utilisatrice est nécessaire.

5. https://www.w3.org/Style/CSS/Overview.en.html

## 3.2 Comparaison OWL/YAMS

Comme détaillé dans la table 2, YAMS est moins expressif que les profils OWL. Il ne couvre pas entièrement le fragment $\mathcal{AL}$ de la logique de description, car il ne permet pas d'exprimer une intersection de concepts. Il s'éloigne également de OWL-Lite par l'absence de transitivité des rôles et leur hiérarchie.

Certains fragments de la logique de description sont toutefois partiellement couverts par YAMS comme la négation, la disjonction de concepts, le "un-de" ou la quantification existentielle typée. Ces expressions sont possibles en YAMS uniquement dans la définition du domaine ou du co-domaine (*range*) d'un rôle.

Comme OWL-Lite, il permet d'exprimer des restrictions de cardinalité pour les valeurs 0 et 1 uniquement [6].

# 4 OWL2YAMS

CubicWeb permet la négociation de contenu et l'administration de données basées sur un modèle de données contenant des connaissances métiers exprimés en YAMS. Afin de s'intégrer dans l'environnement des standards du W3C, nous avons développé un module de traduction

6. Pour une description détaillée des primitives YAMS-OWL, voir https://forge.extranet.logilab.fr/open-source/SemWeb/cubicweb_W3C_standard/-/blob/branch/default/yams_owl.csv

| Logique de Description | formule | YAMS | OWL-Lite | OWL-DL | OWL2 Full |
|---|---|---|---|---|---|
| $\mathcal{AL}$ | $C$ | | | | |
| $\mathcal{AL}$ | $\top$ | | | | |
| $\mathcal{AL}$ | $\forall R.C$ | | | | |
| $\mathcal{AL}$ | $C_1 \sqcap C_2$ | | | | |
| $\mathcal{F}$ | | | | | |
| $\mathcal{U}$ | $C_1 \sqcup C_2$ | | | | |
| $\mathcal{C}$ | $\neg C$ | | | | |
| $\mathcal{AL}$ | $R$ | | | | |
| $\mathcal{E}$ | $\exists R.C$ | | | | |
| $\mathcal{H}$ | $R_1 \sqsubseteq R_2$ | | | | |
| $\mathcal{R}^+$ | $R^+$ | | | | |
| $(\mathcal{D})$ | | | | | |
| $\mathcal{I}$ | $R^-$ | | | | |
| $\mathcal{O}$ | $\{a_1, \ldots, a_n\}$ | | | | |
| $\mathcal{B}$ | $\exists R.\{a\}$ | | | | |
| $\mathcal{N}$ | $(\geq n\ R)$ ou $(\leq n\ R)$ | | | | |
| $\mathcal{R}$ | $R_1 \sqcap R_2$ | | | | |
| $\mathcal{Q}$ | $(\geq n\ R.C)$ ou $(\leq n\ R.C)$ | | | | |

TABLE 2 – Comparaison d'expressivité entre YAMS et les profils OWL (Lite, DL, Full)

OWL2YAMS [7], qui permet de créer une application CubicWeb à partir d'une ontologie exprimée en OWL.

Ce script se base sur une traduction en YAMS d'une ontologie OWL. Comme présenté dans la section 3.2, YAMS est moins expressif que OWL. Tous les axiomes de l'ontologie ne seront donc pas traduits par ce script. Les cases en vert de la colonne YAMS dans la Table 2 ainsi que l'union de concepts dans la définition des domaines et co-domaines des rôles peuvent être traduits par ce script. Les restrictions de cardinalité (limitées à 0 ou 1 dans YAMS) ne sont pas prises en compte pour l'instant.

La figure 2 présente le fonctionnement général de OWL2YAMS. Chaque *owl:Class* ou *rdfs:Class* de l'ontologie est transformée en type d'entité dans le modèle YAMS. Les *owl:DatatypeProperty* sont transformées en attributs en s'assurant que le *rdfs:domain* correspond au type d'entité YAMS et le *rdfs:range* est utilisé avec une correspondance entre les types principaux de *XSD* et les types YAMS. Les *ObjectProperty* sont transformées en relations entre les types d'entités.

Une table de correspondances permet de conserver les URI des éléments de l'ontologie dont sont issus les éléments du schéma YAMS. Cette table est utilisée plus tard pour l'import de données RDF et pour leur traduction en RDF une fois intégrées à CubicWeb (c.f. section 5).

Le script OWL2YAMS crée le schéma YAMS, la structure du cube et l'instance de l'application prête à être lancée et peuplée.

# 5 Import RDF

Une fois l'application créée, il est possible d'y importer des données tant que ces données sont décrites avec l'ontologie ayant servi à créer cette application. La table de correspondances entre les URI de l'ontologie et les éléments YAMS permet de faire le lien entre données RDF et instance CubicWeb.

La figure 3 schématise l'import de données RDF dans l'application créée à partir de l'ontologie OWL.

De nouvelles URI propres à cette instance CubicWeb sont créées pour chaque individu importé. Les URI originales sont conservées dans la base de données. Ces données sont modifiables par l'utilisateur dans l'interface graphique de CubicWeb.

Les données présentes dans CubicWeb sont téléchargeables en RDF par négociation de contenu via les URI propres à l'instance CubicWeb. La figure 4 reprend l'exemple précédent avec les données d'entrée et celles qui seront renvoyées par la négociation de contenu.

Dans l'exemple, l'URI *http ://<MY-CUBICWEB-BASE-URL>/FoafPerson/123* a été associée à l'URI en entrée *http ://example.org/virginia*. C'est sur *http ://<MY-CUBICWEB-BASE-URL>/FoafPerson/123* que la négociation de contenu sera rendue possible. Un triplet *owl:sameAs* permet de rallier l'URI de l'individu dans l'application et son URI d'origine. Nous avons également pris le parti d'exprimer les éléments du schéma YAMS avec leur propre URI (celle attribuée par l'instance CubicWeb). En effet, les éléments du schéma YAMS ne sont pas strictement égaux à ceux de l'ontologie OWL dont ils découlent, par souci de différence d'expressivité. Pour ne pas perdre la sémantique, nous avons choisi d'exprimer le lien entre ces éléments YAMS et OWL avec une subsomption.

# 6 Conclusion et perspectives

Nous proposons dans cet article une méthode pour générer une application Web à partir d'une ontologie OWL et

7. https://forge.extranet.logilab.fr/cubicweb/owl2yams/

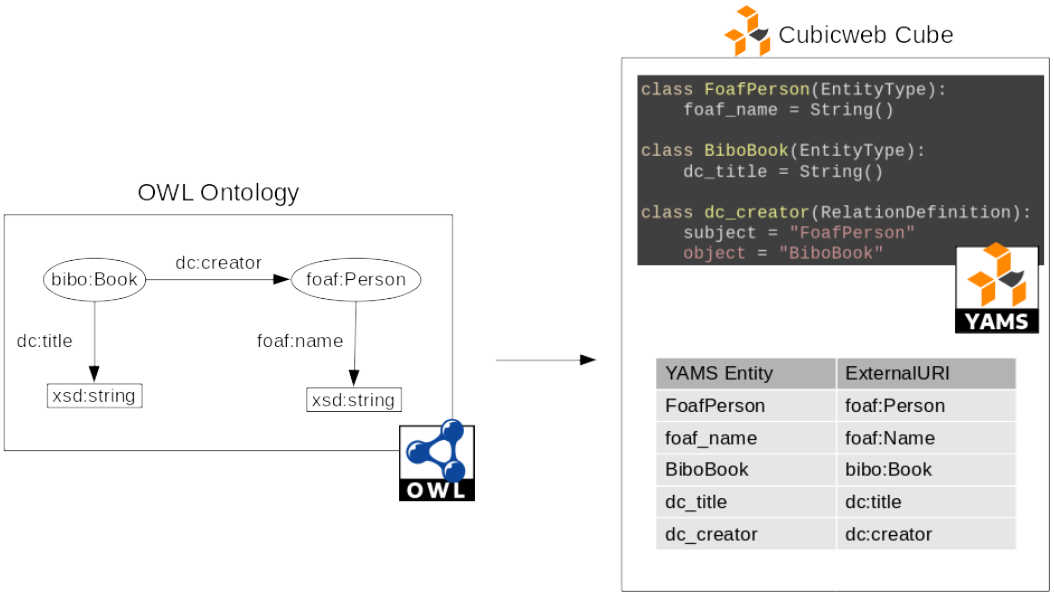


FIGURE 2 – Fonctionnement de OWL2YAMS

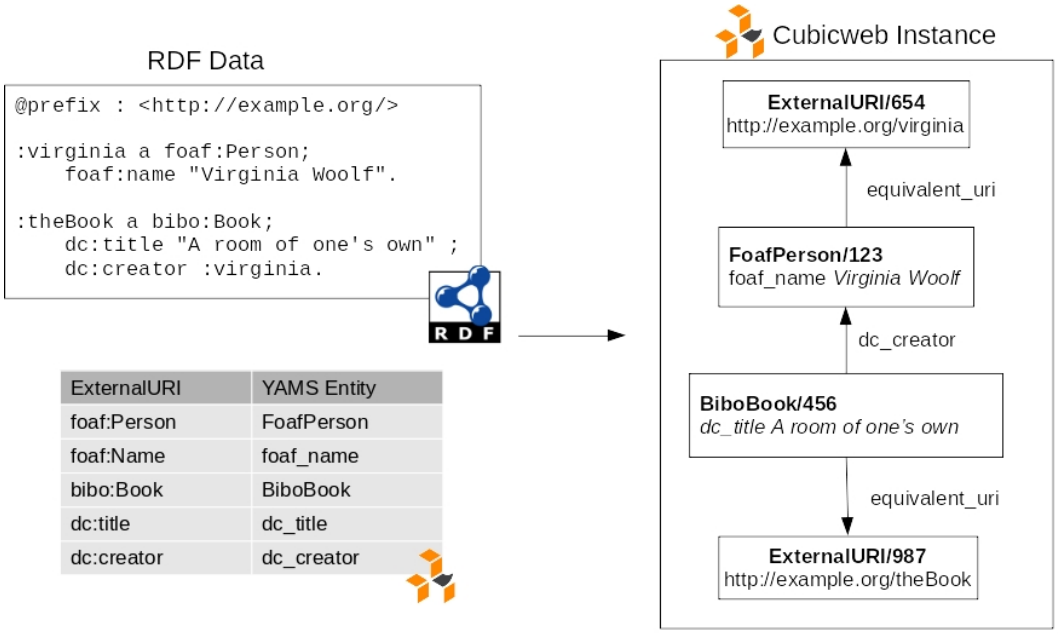


FIGURE 3 – Import de données RDF dans l'application CubicWeb

de données RDF en utilisant OWL2YAMS pour créer un cube et une instance de CubicWeb. De cette manière, il devient très simple de publier des données et l'ontologie associée sur le Web de données liées, en profitant d'un cadriciel complet. CubicWeb propose des fonctionnalités attendues comme la négociation de contenu, la gestion fine des permissions, une interface d'administration, une interface de visualisation et la possibilité de créer sa propre application *front-end* en exploitant les données provenant de CubicWeb.

OWL2YAMS souffre encore de quelques limitations que nous allons lever. À ce stade, certains prédicats de OWL-Lite ne sont pas pris en compte même alors que YAMS permettrait de les exprimer. C'est notamment le cas pour la définition multiple d'attribut (actuellement un attribut ne pourra être défini qu'une seule fois par modèle, par exemple un seul *rdfs:label*), l'héritage multiple de classe ou encore les cardinalités. OWL2YAMS est à ce stade une preuve de concept que nous devrons continuer à développer avant de pouvoir l'utiliser dans des projets en production.

Par ailleurs, dans le but de permettre l'interrogation de CubicWeb avec SPARQL, nous avons étudié les différences d'expressivité entre RQL et SPARQL. Sachant que RQL est transformé en SQL, qui repose sur un paradigme de monde fermé, alors que SPARQL repose sur un paradigme de monde ouvert, la transformation directe de SPARQL en

```
:virginia a foaf:Person;
    foaf:name "Virginia Woolf".

:theBook a bibo:Book;
    dc:title "A room of one's own" ;
    dc:creator :virginia.
```

**a) Données RDF en entrée**

```
cw:BiboBook rdfs:subClassOf bibo:Book .
cw:FoafPerson rdfs:subClassOf foaf:Person .

cw:dc_author rdfs:subPropertyOf dc:author .
cw:dc_title rdfs:subPropertyOf dc:title .
cw:foaf_name rdfs:subPropertyOf foaf:name .

cw:FoafPerson/123 a cw:FoafPerson ;
    cw:foaf_name "Virginia Woolf" ;
    owl:sameAs <http://example.org/virginia> .

cw:BiboBook/456 a cw:BiboBook ;
    cw:dc_title "A room of one's own" ;
    cw:dc_creator cw:FoafPerson/123 ;
    owl:sameAs <http://example.org/theBook> .
```

**b) Données RDF en sortie**

FIGURE 4 – Exemple de données RDF en entrée et en sortie de l'application

RQL nous semble difficile. Jusqu'à maintenant nous avons dupliqué les données de CubicWeb dans un entrepôt RDF adjacent chaque fois que l'interrogation en SPARQL était requise. Nous allons continuer à explorer cette question.
Pour finir, notre objectif à moyen terme est de proposer CubicWeb *as a service*, afin que des personnes souhaitant publier des données puissent y parvenir en déposant une ontologie OWL et un graphe RDF dans un formulaire dont la validation déclencherait la création d'une application CubicWeb et son déploiement sur un cluster Kubernetes [8]. Ceci mettrait à la portée de toutes et tous l'intégration de connaissances au Web des données.

# 7 Bibliography

# Références


[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 34–43, 2001.

[2] A. Simon, R. Wenz, V. Michel, and A. D. Mascio, "Publishing bibliographic records on the web of data : Opportunities for the bnf (french national library)," in *ESWC*, ser. Lecture Notes in Computer Science, vol. 7882. Springer, 2013, pp. 563–577.

[3] F. Amarger, S. Chabot, N. Chauvat, and E. Thiéblin, "Cubicweb : vers un outil pour des applications clé en main dans le web sémantique," in *31es Journées francophones d'Ingénierie des Connaissances*, 2020.

[4] T. Heath and C. Bizer, "Linked data : Evolving the web into a global data space," *Synthesis lectures on the semantic web : theory and technology*, vol. 1, no. 1, pp. 1–136, 2011.


8. https://kubernetes.io/fr/