Programmer le web de données avec un « Wiki-based IDE »

Pavel Arapov, Michel Buffa, Amel Ben Othmane

Equipe Wimmics, commune aux laboratoires
INRIA et I3S de Sophia Antipolis,
{arapov, buffa, amel.ben-othmane}@i3s.unice.fr

Résumé : WikiNEXT est un wiki à la croisée des wikis sémantiques et des outils de développement en ligne récents (« web based IDEs »). En permettant de coder directement dans le navigateur des applications exploitant le web de donnée et manipulant des données sémantiques, WikiNEXT étend le concept de wiki sémantique et répond au problème *« comment faciliter la programmation et l'apprentissage d'applications pour le web sémantique ? »*. WikiNEXT s'adresse à plusieurs profils d'utilisateurs, cependant cet article se focalise sur les aspects « wiki programmable » qui concernent principalement les développeurs web. Les wikis sémantiques actuels permettent d'insérer du contenu dynamique dans les pages, mais ne partagent pas cette approche, nous montrerons en quoi WikiNEXT améliore l'état de l'art dans le domaine. L'outil est en ligne sur http://wikinext.gexsoft.com et contient de nombreux tutoriaux interactifs.

Mots-clés : wikis, web-IDE, wikis sémantiques, JavaScript.

1 Introduction

Le problème général auquel WikiNEXT essaie de répondre est *"comment programmer plus facilement des applications exploitant le web de données et les données sémantiques ?"*

Par exemple, nous voulons un graphique de la population des pays n'ayant pas de frontière maritime, et ayant plus de 1.500000 habitants, à partir de données de DBpedia.org (Figure 1).

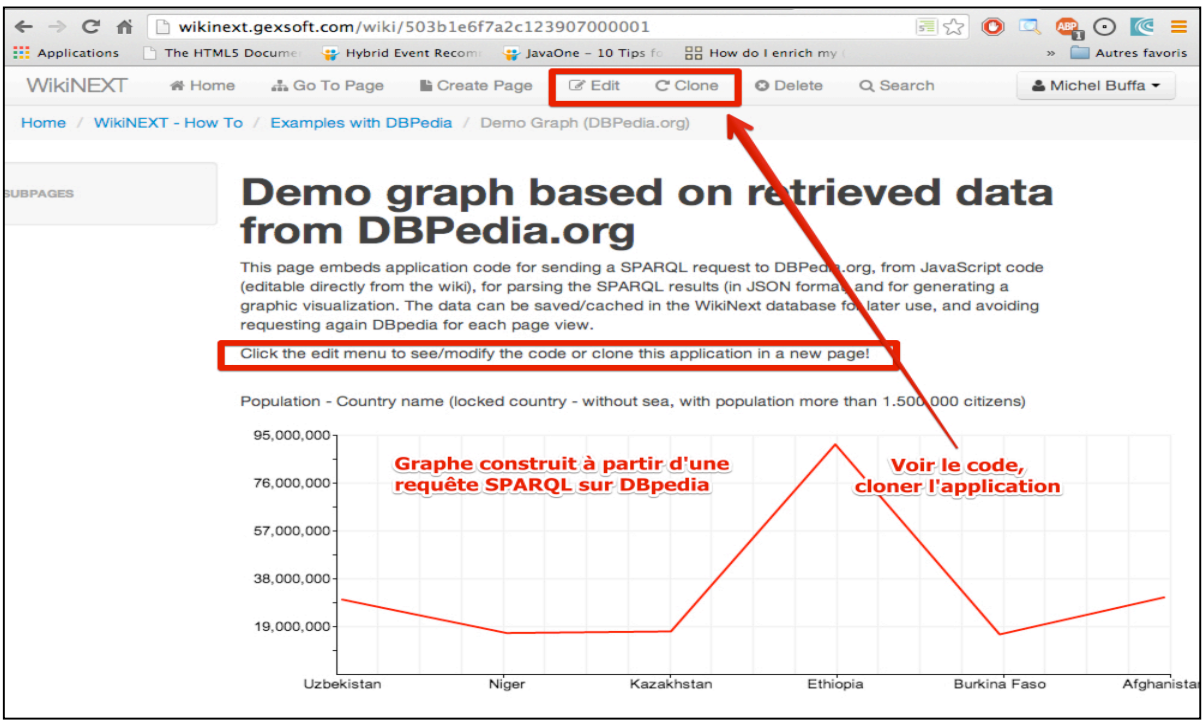


Figure 1 Exemple de contenu dynamique dans WikiNEXT

… on pourra utiliser l'éditeur JavaScript/HTML intégré, qui permet de voir le code de l'application, ou encore, on pourra cloner l'application pour l'examiner et la modifier. Cette approche par mimétisme est caractéristique des wikis et demande peu d'expertise (Figure 2).

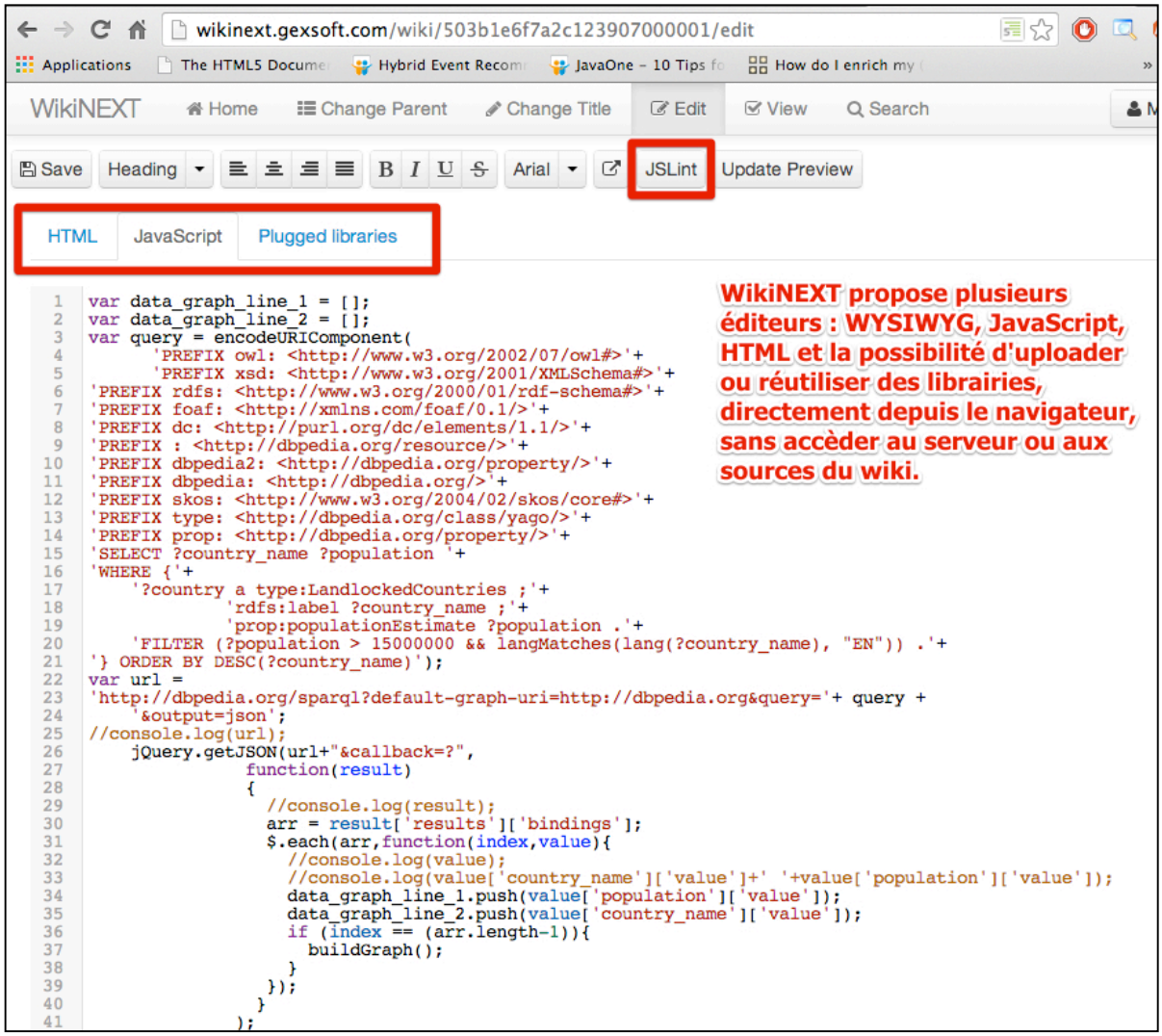


Figure 2 : l'envers du décors de l'application de la Figure 1.

WikiNEXT se programme directement dans le browser et exploite les tendances récentes des outils de développement basés web rendues possible par les avancées de HTML5 et de JavaScript. En proposant des fonctionnalités de clonage et en mélangeant document et application, l'apprentissage par mimétisme se retrouve facilité. WikiNEXT est en développement depuis trois ans (Arapov et Al. 2012), et a été écrit à partir de zéro. Les motivations viennent d'un constat d'expérience : nous avons déjà développé deux wikis sémantiques depuis 2006 : SweetWiki (Buffa et Al. 2008) et DekiWiki (Buffa et Al. 2012), et nous nous sommes heurtés aux limitations que partagent aussi les wikis sémantiques les plus populaires : ils ne permettent pas facilement d'écrire des applications exploitant les données sémantiques et le web de données. La raison principale est le choix d'une approche « par macros » ou « par extensions » au lieu de proposer une interface de programmation dans le navigateur. Pour réaliser l'application présentée, on aurait avec les wikis actuels, inséré une macro dans un document, que l'on aurait pu paramétrer (pour indiquer la requête, le type de graphe, etc.), mais le code « derrière » une macro nécessite des outils *ad hoc* lourds et un accès au serveur. Seuls des experts peuvent s'attaquer à ce type de développement.

L'originalité de WikiNEXT est qu'il est à la fois un wiki sémantique, et un IDE[1] (Integrated Development Environment), il considère les pages comme des applications contenant du texte, des images ou autres médias, des métadonnées et aussi une partie programmable. Les avantages par rapport à l'approche basée « macros » sont nombreux :

- *Programmation instantanée* (« live coding ») : je code et je vois le résultat quasiment en temps réel de mes modifications, cette approche facilite les expérimentations. Elle est très adaptée à des applications pédagogiques,
- *Partage des applications créées dans le wiki* (clonage, partage),
- *Documentation « sur place » des applications* : une page du wiki peut contenir un tutorial ou la documentation de sa partie applicative,
- *Utilisation de langages standards (JavaScript, JSON, RDF, RDFa, SPARQL, etc.).*

WikiNEXT propose également des outils pour aider le développeur : il fournit une riche API accessible depuis le code JavaScript, des aides interactives (pour la création de requêtes SPARQL par exemple), un système de templates pour l'affichage de données sémantiques, une base de données de graphes compatible SPARQL 1.1 pour le raisonnement et pour l'implémentation d'un SPARQL endpoint, une base de données objet pour la persistance, etc.

L'outil est en ligne[2], open source, et propose de nombreux exemples et tutoriaux montrant comment collecter des données depuis DBPedia, depuis Freebase, ou depuis la base de connaissance locale du Wiki. Les démonstrations et tutoriaux sont variés : mashup de plusieurs sources de données, annotation automatique de documents, génération de vues personnalisées sous forme de page de wiki ou sous forme de visualisations graphiques (templates), utilisation d'un cache permettant de ne pas requêter le web de données à chaque affichage de page enrichie, avec gestion de la durée de vie des données, etc.

WikiNEXT s'adresse à plusieurs profils d'utilisateurs. Un utilisateur lambda peut utiliser l'éditeur WYSIWYG et créer des documents classiques, cependant nous avons mis dans cet article l'accent sur le côté « wiki pour écrire des applications ». Nous nous adressons donc aux développeurs d'application web et web sémantique, mais aussi aux enseignants et à ceux qui utilisent déjà des petits « web IDEs » pour tester du code.

Etat de l'art : convergence entre wikis classiques, wikis d'application, wikis sémantiques et « environnement de développement basés web »

Wikis d'application : Issus du Web 2.0, les wikis, dont le représentant le plus illustre est WikiPedia, sont connus pour favoriser la convergence de l'écriture collaborative de documents. Rapidement, certains moteurs de wikis ont étendu cette approche pour aller au-delà de l'écriture de simples documents textuels. Les « wikis d'application », aussi appelés « wikis d'entreprise », ont ajouté le moyen de « programmer dans les pages du wiki ». TWiki (encore populaire de nos jours), en 1998, a été un pionnier de cette approche en proposant d'utiliser des macros pour générer du contenu dynamique dans les pages. Par exemple on pouvait générer des tableaux, des formulaires, des champs des recherche dans les pages, les données manipulées étant elles aussi stockées dans des pages ayant une structure particulière (sous forme de tables ou de liste HTML). En 2005, d'autres wikis, comme JOT (devenu par la suite Google Sites), XWiki ou Mindtouch Core[3] sont allés plus loin en intégrant de véritables langages de script. Ainsi des utilisateurs avancés ayant des notions de programmation pouvaient développer des « petites applications dans le wiki » incluant la définition de modèles de données, leur traitement et leur présentation. Tout en conservant la possibilité pour d'autres utilisateurs de copier/coller une page/application et de modifier légèrement le travail cloné pour en faire une version personnelle. C'est ce que Ward Cunningham appelle *« the Wiki Way »* et qui transforme les wikis en puissants outils pour *« apprendre en*

[1] IDE = Integrated Development Environment. Eclipse ou Netbeans sont des IDEs classiques, jsbin.com, jsfiddle.net sont des IDEs en ligne, ou « web based IDEs ».

[2] http://wikinext.gexsoft.com.

[3] http://www.mindtouch.com/, utilise DekisScript, un langage inspiré par JavaScript, pour écrire du code dans les pages.

regardant le travail des autres » (Leuf et Cunningham 2001). Le code source de ces applications et les données ne sont pas visibles par les utilisateurs qui consultent les pages, mais éditer une page révèle les secrets de fabrication (le script, les macros) de l'application qu'elle renferme. Ces macros sont soit prédéfinies, soit écrites par les utilisateurs à l'aide d'un langage de script (Velocity[4] ou Groovy[5] pour XWiki, DekiScript, pour le wiki Mindtouch Core, etc.) Les ensembles de macros peuvent utiliser des plugins ou des extensions ajoutés côté serveur (par exemple un plugin pour se connecter à une base de données SQL externe, fournira une macro utilisable dans les pages). *Dans ce cas, le développeur du plugin ou de l'extension doit avoir accès au code source du Wiki, ou à un SDK fourni avec le moteur de wiki. Parfois le nombre d'outils nécessaires pour pouvoir développer un plugin est très important.*

Les Wikis sémantiques : inspirés par les wikis traditionnels, les chercheurs ont commencé à partir de 2005 à proposer des « wikis sémantiques » tels que Semantic Media Wiki (Krötzsch et Vrandecic 2006), IkeWIki (Shaffert 2006) ou encore SweetWiki (Buffa et al. 2008), voir (Buffa et al. 2008) et (Krotzsh et al. 2007) pour une synthèse des moteurs de wiki sémantiques de la première génération (2005-2008). Ces wikis permettent à leurs utilisateurs d'ajouter du contexte sémantique aux documents tout en préservant la simplicité et l'essence collaborative des wikis. Les ressources sémantiques sont en général conservées en interne (dans une base de connaissance) sous la forme de données RDF, OWL ou sous forme de graphes conceptuels (Oren, Breslin, Decker 2006). Les données sémantiques produites sont déjà intéressantes en elles-mêmes (par exemple dans le cas de production collaborative d'ontologie, ou d'annotation de documents), dans la mesure où elles sont naturellement difficiles à produire, mais surtout, les wikis équipés de moteurs sémantiques peuvent les exploiter pour du raisonnement, implémenter des outils de recherche augmentée, des systèmes de suggestion, etc. Les wikis sémantiques proposent souvent des langages intermédiaires proches du langage naturel (on écrit « this is a class » dans Semantic Media Wiki pour créer une classe qui sera un concept dans une ontologie) ou à base de langage de markup popularisé par la « culture wiki ». Semantic Media Wiki, ou SemperWiki (Oren 2005) utilisent une syntaxe à base de crochets pour intégrer des annotations, qui seront ensuite traduites en RDF une fois la page sauvegardée. Cette approche, très simple, basée sur les usages des wikis de l'époque, a posé quelques problèmes de cohérence dans les premières générations de wikis sémantiques (2005-2010). IkeWIki (Shaffert 2006) a combiné l'approche classique des wikis pour la création de documents avec une interface à base de formulaires avec auto-complétion pour faciliter la réutilisation de vocabulaires et la vérification de cohérence. OntoWiki (Auer et al. 2006) quant à lui est un véritable éditeur d'ontologie avec une interface utilisateur proposant différentes vues pour naviguer et organiser l'ontologie produite. SweetWiki (Buffa et al. 2008) proposait un système de tagging social sémantique et conceptualisait le wiki lui-même sous la forme d'une « ontologie du wiki », implémentant ainsi la quasi-totalité de ses fonctionnalités autours des technologies du web sémantique.

Aujourd'hui, peu de wikis sémantiques sont encore maintenus[6] on pourra retrouver un état de l'art récent dans (Meilendera et al. 2010). Existent encore des wikis de l'ancienne génération qui ont (1) étendu leurs fonctionnalités en devenant des hybrides de CMS (Content Management Systems) ou (2) au contraire en se « coupant en morceaux » et en externalisant certaines fonctionnalités à des entités externes dédiées. Dans la première catégorie nous retrouvons Semantic Media Wiki avec aujourd'hui de très nombreuses extensions, comme Halo, qui propose des formulaires d'annotation avec auto-complétion, un éditeur WYSIWYG, l'intégration de fichiers multimédia et la mise en place d'un « SPARQL endpoint ». On trouve aussi des wikis basés sur Semantic Media Wiki, comme Moki (Rospocher et al 2009),

[4] http://velocity.apache.org/

[5] http://groovy.codehaus.org/

[6] Il y en avait 37 de recensés lors du workshop qui leur était dédié lors de la conférence ESCW 2006, moins d'une dizaine sont encore actifs en 2013.

spécialisé sur la modélisation de processus d'entreprises, ou OWiki (Di Lorio et al. 2010), dédié à la génération de contenu dirigé par des ontologies. Dans la seconde catégorie on trouve le projet KiWi -Knowledge In Wiki- (Shaffert et al. 2008), qui implique les auteurs de IkeWiki, refondu dans un projet comprenant de nombreux modules, ou SweetDeki (Buffa et al. 2013, Buffa et Husson 2012), le wiki qui a succédé à SweetWiki, fondu dans le projet ANR ISICIL (Gandon et al. 2009) intégrant un réseau social, un serveur de tags, un gestionnaire de ressources sémantiques externes, etc. Ces projets ont essayé de mettre le wiki sémantique dans un contexte industriel (ISICIL a donné naissance à la société Mnemotix.com, KiWi est appuyé par des fabricants de CMSs). Bien que les wikis sémantiques cités permettent la création d'ontologie ou l'annotation de documents, l'exploitation des données sémantiques par l'utilisateur se limite souvent à l'insertion de requêtes ou macros dans les pages, faisant références à des extensions ou plugins éventuellement installés. *Dans ce sens, ces wikis sémantiques modernes ont convergé vers les wikis d'applications « historiques » présentés dans la précédente section, et partagent également leurs défauts : développer un nouveau plugin ou une extension requiert un SDK ou l'accès aux sources du wiki. On peut développer de petites applications « dans le wiki », mais avec des contraintes importantes.*

Outils de développement basés web (*« Web-based IDEs »*) : L'idée consistant à écrire des applications directement dans un navigateur web n'est pas récente : les premières expérimentations datent de 1996 (Crespo et al. 1996). Depuis 2005, avec la technologie Ajax et le développement de JavaScript, les éditeurs de code source pouvant fonctionner dans une page web n'ont cessé de s'améliorer. Des éditeurs comme Code Mirror[7], ACE Cloud9[8] or ternjs[9] proposent aujourd'hui la colorisation de syntaxe, l'auto-complétion, des fonctions d'édition complètes, et équipent plusieurs sites web connus proposant des environnements de développement basés web[10]. Certains supportent plusieurs langages de programmation comme compile-online.com, ideone.com ou compilr.com, mais les plus intéressants sont sans doute les environnements dédiés à la programmation JavaScript/HTML/CSS, très appréciés des *développeurs web*. Des sites comme jsbin.com, jsfiddle.net, codepen.io ou tinkerbin.com proposent d'écrire du code directement dans le navigateur et de voir l'exécution en temps réel, pendant la saisie. On parle de « live coding », ou « codage vivant », qui est particulièrement appréciable pour évaluer rapidement des algorithmes, des idées, ou tout simplement pour tester des fonctions d'APIs de HTML5. Ces outils proposent souvent une option de clonage d'une application existante. La plupart de ces outils, cependant, ne sont que des bacs à sable pour tester du code ou pour écrire des exemples à but pédagogique. Enfin, on trouve des outils de développement comme cloud9ide.com, shiftedit.net, coderun.com ou codeanywhere.net qui ambitionnent de remplacer des outils comme Eclipse, en proposant de gérer en ligne le développement de véritables projets composés de multiples fichiers JavaScript/HTML/CSS. WikiNEXT reprends certaines fonctionnalités de la dernière génération des outils présentés dans cette section. A notre connaissance, aucun wiki sémantique n'a encore adopté une telle approche pour « programmer dans le wiki ».

2 WikiNEXT

Motivation : wikis d'application, wikis sémantiques, IDEs basés web : WikiNEXT est un hybride, un mélange de ces trois approches. Tout d'abord, dans WikiNEXT, *chaque page est une application web versionnée*, composée de code JavaScript/HTML/CSS. Nous l'avons vu dans l'introduction, plusieurs éditeurs sont fournis : un éditeur WYSIWYG pour des documents classiques, et un éditeur de code pour la partie « application ». Par ailleurs WikiNEXT expose une API donnant aux applications accès aux mécanismes internes du wiki,

[7] http://codemirror.net/

[8] http://ace.c9.io/

[9] http://ternjs.net/

[10] See http://en.wikipedia.org/wiki/Comparison_of_JavaScript-based_source_code_editors

et facilitant la manipulation de données sémantiques (requêtes, affichage, persistance). Considérons par exemple une page classique créée par un utilisateur ne sachant pas programmer. Il pourra demander à une personne ayant des compétences en développement web « d'augmenter sa page » en écrivant un bout de code qui collecte des informations d'une source de données externe comme DBPedia afin d'annoter le document qu'il a écrit, ou pour extraire des éléments du texte afin de suggérer une classification, insérer une visualisation graphique, ou sauvegarder les données collectées dans la base de connaissances du wiki. Il sera possible par la suite de cloner la page/application existante pour l'étudier et en faire une version différente. Comme dans tout wiki, il est nécessaire d'amorcer le wiki en proposant une base de données initiale d'exemples variés, de tutoriaux, afin de faciliter l'apprentissage par clonage/modification. WikiNEXT propose des services pour l'annotation de textes, la persistance « back end », côté serveur, en fournissant notamment une base de données de graphe compatible RDF/SPARQL 1.1, et une base de données objet, WikiNEXT est aussi un SPARQL endpoint. Nous avons mis l'accent sur cet aspect « programmation » du wiki, car les applications sémantiques sont souvent associées avec des « documents » qui sont soit sources de données, soit cibles pour des annotations ou des enrichissements. Par ailleurs, WikiNEXT a aussi une vocation pédagogique, en tant qu'outil permettant de développer des tutoriaux interactifs. On peut, par exemple, écrire un document avec du texte explicatif, mélangé avec l'application qu'il décrit : un tutorial sur « comment requêter DBPedia » contiendra à la fois le texte et le code de l'application, présentera les résultats d'exécution en temps réel et proposera de modifier directement le code pour voir le résultat des modifications.

La couche « application » d'un document est constituée de code JavaScript qui peut être édité de la même manière que l'on peut éditer le code HTML de la page. Ainsi il est possible de manipuler depuis le code JavaScript le Document Object Model (DOM) de la page comme on le fait dans la programmation JavaScript classique : rajouter un id sur un HTML pour afficher des résultats, ou bien utiliser le système de templates de présentation de WikiNEXT, présenté dans la section suivante. Certaines tâches comme « créer une nouvelle page du wiki », « requêter la base de connaissance du wiki », « annoter un document et maintenir les vues sur ces annotations à jour lorsque les valeurs changent » (liste non exhaustive) sont réalisées par des fonctions de l'API interne du wiki, alors que d'autres peuvent s'appuyer sur des librairies JavaScript externes telles que D3.js[11], une puissante librairie graphique pour la visualisation de données. WikiNEXT propose par ailleurs un outil intégré permettant d'uploader des librairies externes et de sélectionner celles utilisées par chaque page/application. Le prototype est open source[12] et disponible en ligne[13]. Il inclut aujourd'hui de nombreux exemples et tutoriaux. Nous l'utilisons avec des étudiants de Master pour illustrer certaines utilisations du web de données et pour apprendre les langages RDF/SPARQL.

Architecture du logiciel : dans le paradigme de développement agile, des chercheurs ont montré que les wikis peuvent être utilisés comme plate-forme pour l'échange d'objets logiciels légers réutilisables (Rech, Bogner et al. 2007), cependant il s'agissait de « bouts de logiciels » externes au wiki. WikiNEXT reprend ces principes et les mets en œuvre dans le paradigme MVC, la partie JavaScript de la page formant « la couche métier et le contrôleur », la partie « vue » se faisant dans le code HTML / CSS de la page, et la partie « modèle » étant composée de métadonnées RDF. La synchronisation de la vue et des modèles est implémentée par WikiNEXT. Il est également possible de synchroniser les modèles avec les bases de données de WikiNEXT, situées côté serveur, via l'API WikiNEXT. Chaque page WikiNEXT est associée avec un ensemble de métadonnées qui décrivent ses principales caractéristiques : titre, auteur, contributeurs, dernière date de modification, version, etc. Mais les pages sont

[11] http://d3js.org

[12] https://github.com/pavel-arapov/wikinext

[13] http://wikinext.gexsoft.com

aussi des « containers » : elle contiennent des métadonnées qui auront pu être rajoutées manuellement ou par une application d'annotation. WikiNEXT s'appuie sur les ontologies de schema.org[14] pour décrire la structure des pages, les utilisateurs, etc. Chaque page est l'instance d'un Article, elle est représentée par un graphe nommé constitué d'annotations RDFa/RDFAlite[15], dont le nom est basé sur l'URI de la page (Figure 3). Nous appelons l'ensemble des métadonnées localisées dans une page la *base de connaissances locale*. L'ensemble des pages du wiki, est appelé *base de connaissances globale*, elle est constituée de l'union de tous les graphes nommés. Les métadonnées générées par la partie programmable d'une page, par exemple une liste de pays provenant d'une requête sur DBPEdia.org, pourront être injectées dans la page sous forme d'annotations RDFa/RDFalite. Elles seront également sauvegardées dans un triple store intégré à la partie serveur du Wiki lors de la sauvegarde de la page.

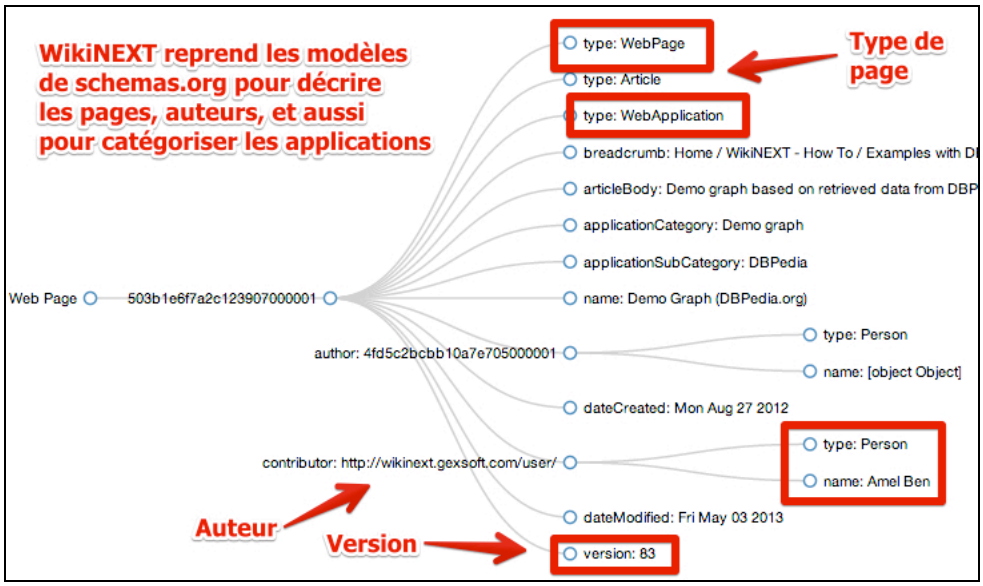


Figure 3: Le modèle des pages de WikiNEXT

Implémentation (Figure 4**) :** WikiNEXT a été écrit « from scratch » en JavaScript. Il s'appuie sur le micro-serveur Node.js[16] et sur la base de données MongoDB[17]. Node.js embarque l'interpréteur JavaScript V8 de Google, qui équipe entre autre le navigateur Google Chrome. Ainsi, Node.js permet ce que l'on appelle le « server-side JavaScript », une tendance émergente dans le développement web. MongoDB est une base de données NoSQL orientée documents (indexation du contenu, etc.). Au lieu de stocker des objets sérialisés sous forme de lignes et colonnes dans des tables comme on le ferait avec une base de données relationnelle traditionnelle, ici on stocke des objets JavaScript au format BSON (une forme binaire du format de sérialisation JSON[18] de JavaScript). Voir Figure 4. Nous utilisons pour gérer les triplets RDF une version améliorée de RDFStore-js (Hernandez et Garcia 2012), un moteur de graphes compatible SPARQL 1.1, écrit en JavaScript. Le contenu traditionnel de la page est stocké sous forme d'objets dans la base de données MongoDB (notamment, nous générons des indexes pour le contenu textuel de la page). Nous l'utilisons également comme couche de persistance pour RDFStore-js. Ainsi, les triplets sont accessibles à la fois à travers

[14] http://schema.org/ and http://schema-rdfs.org ont été créés par les principaux moteurs de recherche (Google, Microsoft, Yahoo et Yandex). Ils prooposent des vocabulaires partageables RDF/S et pour les microdatas de HTML5, qui couvrent les principaux domaines.

[15] http://www.w3.org/TR/xhtml-rdfa-primer/ et http://www.w3.org/TR/rdfa-lite/

[16] http://nodejs.org

[17] http://mongodb.org

[18] http://www.json.org, a text based notation for JavaScript objects.

des requêtes SPARQL, mais aussi directement via MongoDB. On dispose d'un accès direct aux nœuds du graphe, ce qui est utile pour implémenter de manière très efficace certaines tâches (recherche d'entités nommées, notamment). Pour l'affichage et la visualisation d'annotations sémantiques, WikiNEXT propose un système de templates. Le cas d'utilisation classique consiste à créer des pages WikiNEXT faisant office de modèles de présentation (templates) qui seront réutilisées par d'autres pages, par leur partie applicative. Le principe est simple et permet de découpler l'interface utilisateur de la définition des données.

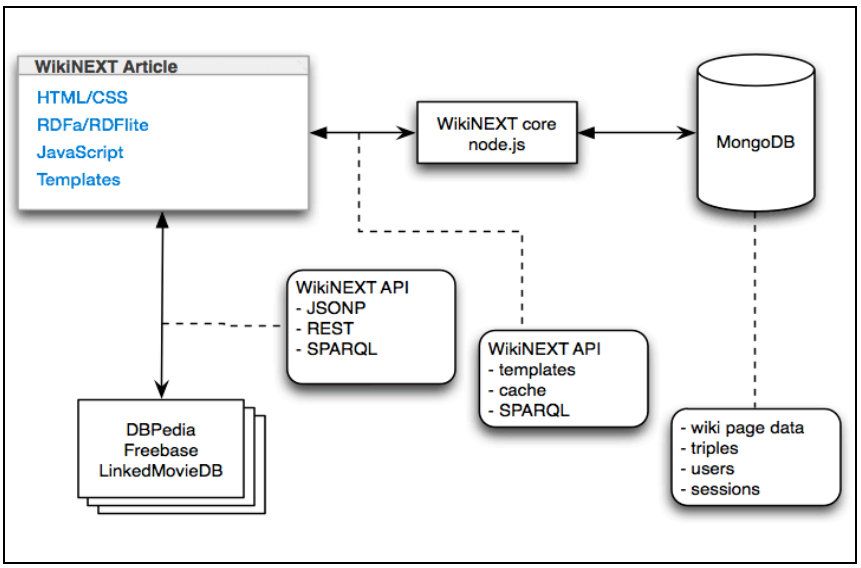


Figure 4: Architecture technologique de WikiNEXT

Les templates, la manipulation de requêtes SPARQL, la manipulation d'annotations RDF, la persistance, le cache des données, toutes ces fonctionnalités sont accessibles par le code des applications que l'on écrit dans le wiki, au travers de l'API WikiNEXT. Avec ces choix technologiques, nous avons essayé de minimiser le nombre de langages de programmation utilisés et le nombre de formats de données. Avec WikiNEXT, JavaScript est utilisé le long de la chaîne complète, depuis le code fonctionnant dans le navigateur web (code client) jusqu'au code serveur et au format de persistance.

3 Mashup sémantique et utilisation de la base de connaissance globale de WikiNEXT

WikiNEXT permet de développer des applications plus complètes et plus variées que l'exemple très simple présenté dans l'introduction. Nous décrivons ici un scénario d'usage complet : le développement d'une application qui utilise le SPARQL endpoint de DBPedia.org pour récupérer des données RDF concernant des villes (description, population, photos, etc.). Les résultats sont utilisés pour créer à la volée des pages du wiki –une par ville-, basée sur un modèle de présentation, lui aussi créé dans le wiki. Les pages sont annotées et les annotations sont sauvegardées dans le triple store RDF du wiki. Le nom des villes à chercher se fait à l'aide d'un formulaire, lui aussi créé dans le wiki ; enfin, une dernière page réalise un « mashup sémantique » en construisant une carte présentant l'ensemble des villes qui ont été récupérées, avec un résumé et des photos de chacune d'entre elles. Cette application est disponible en ligne sur le site de WikiNEXT[19], c'est un des tutoriaux que l'on peut consulter, éditer et dont on peut voir le code, le modifier ou le cloner[20]. Elle est également disponible en

[19] http://wikinext.gexsoft.com/wiki/519e04c580194c4178000001

[20] Il suffit de se connecter au Wiki (inscription traditionnelle ou via identifiants facebook) et d'éditer la page de l'application « City ».

screencast[21] sur YouTube. Le tutorial guide pas à pas le développeur : on commence par créer en HTML un formulaire de saisie (Figure 5) permettant de choisir la ville à rechercher, le pays, la langue. Les paramètres saisis servent à requêter le SPARQL endpoint de DBPEdia.org. Les résultats sont ensuite utilisés pour créer à la volée une page dans le wiki. Pour cette application on ne va pas conserver le modèle original de DBPedia.org pour les villes mais on va décrire les villes en utilisant des propriétés du modèle http://schema.org/city, plus simple (Figure 7). Lorsqu'une page du wiki est sauvegardée (c'est le cas de la page créée à la volée pour chaque ville-résultat, cf Figure 6), les métadonnées sont également ajoutée à la base de connaissance de WikiNEXT (au format RDF). L'intérêt de cet exemple est de montrer l'utilisation des fonctions d'API pour manipuler les annotations sémantiques et pour illustrer l'utilisation de la base de connaissance RDF/SPARQL de WikiNEXT.

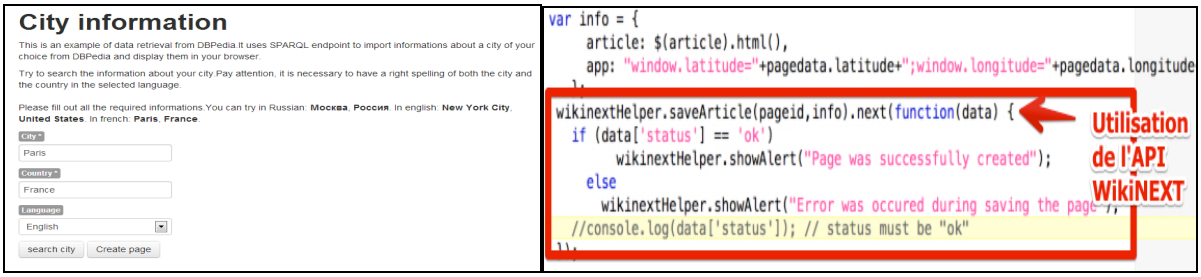


Figure 5: Formulaire de saisie de l'application « City », un des tutoriaux de WikiNEXT, et exemple de code utilisant l'API de WikiNEXT pour créer une page à la volée avec les résultats d'une requête SPARQL.

L'API fournit d'autres fonctionnalités comme la possibilité de cacher/mémoriser des données avec une certaine durée de vie. Par exemple, la page de la ville de Paris contient des annotations concernant la population, mais on peut indiquer que ces données ont une certaine durée de vie. Passé un délai, l'affichage de la page déclenchera un rafraichissement des données depuis DBPedia.org (au lieu de les prendre dans le cache local, c'est-à-dire dans la base de connaissance locale, voir un des exemples en ligne disponibles[22]).

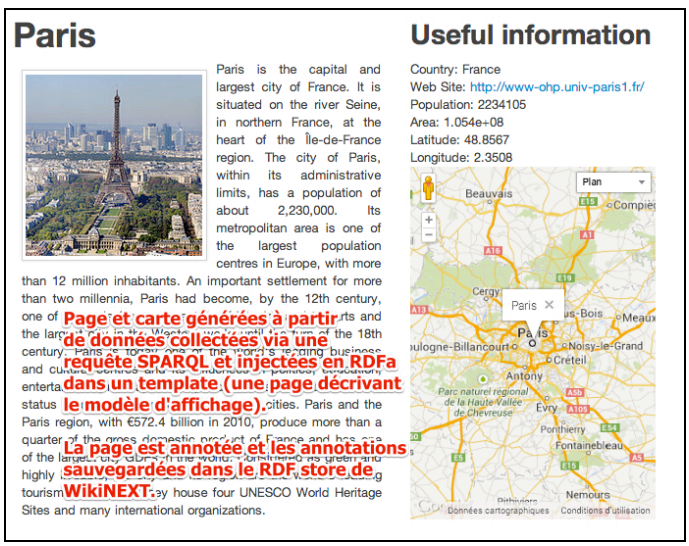


Figure 6: Exemple de page générée

[21] Chercher « WikiNEXT » sur Youtube.com

[22] Exemple d'application utilisant un cache avec durée de vie : http://wikinext.gexsoft.com/wiki/511a4f8207d1b7e451000001

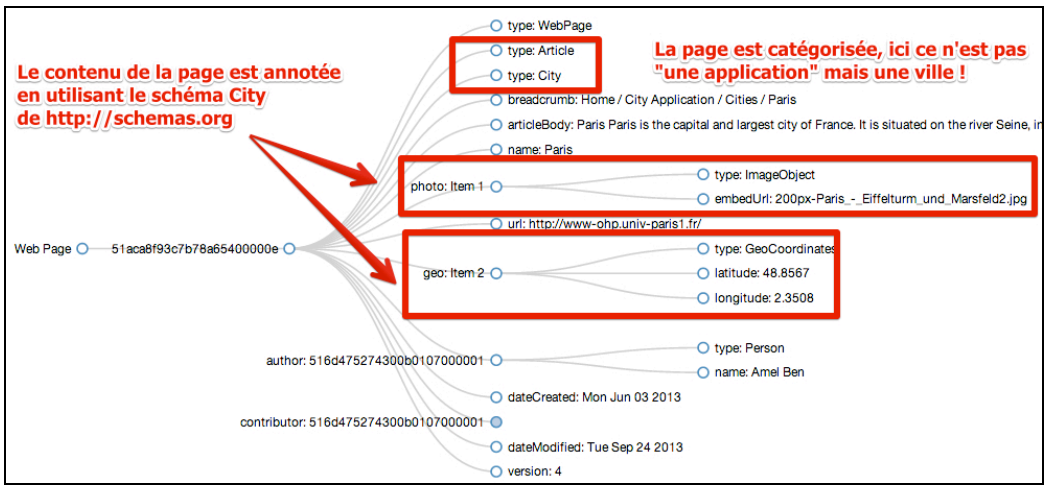
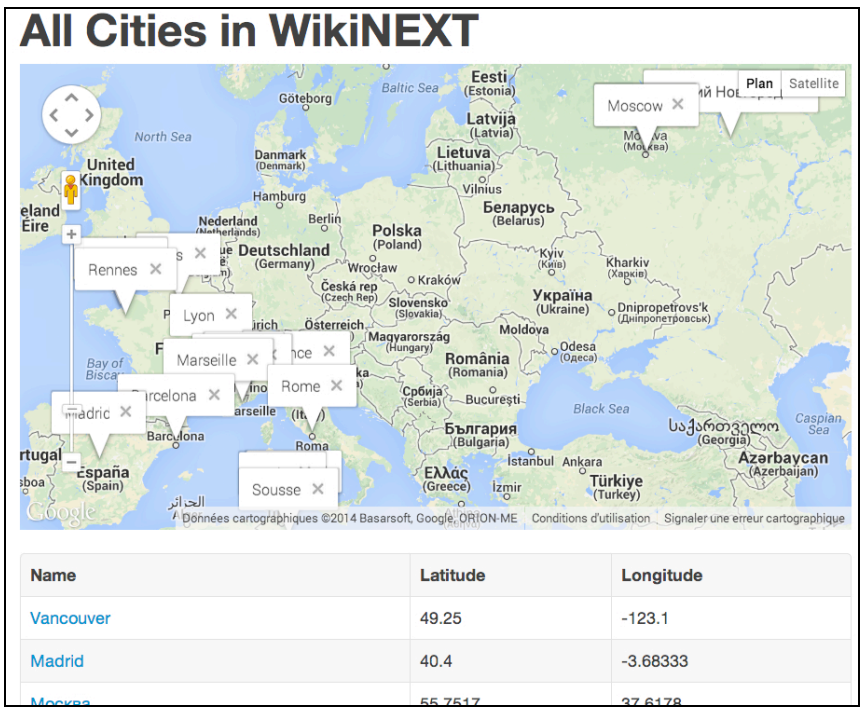


Figure 7: Annotations d'une page créée dynamiquement, pour la ville de Paris.

Réutilisation des données : pour illustrer l'utilisation de la base de connaissance globale de WikiNEXT, nous pouvons également réaliser un mashup à partir des métadonnées associées aux pages des villes construites à partir de requêtes envoyées depuis le formulaire présenté précédemment. Rappelons que les métadonnées correspondant à chaque ville sont à la fois présentes dans les pages mais également dans le triple store de WikiNEXT. Pour l'application de mashup (une carte avec l'ensemble des villes et une liste contenant une vue résumée de chaque ville), nous créons une nouvelle application/page dans le wiki mais cette fois-ci nous requêtons en SPARQL la base de connaissance du wiki au lieu de requêter le web de données, les résultats sont présentés Figure 8. WikiNEXT pré-charge les ontologies de schema.org, les données sur les villes stockées dans WikiNEXT ont été, dans cet exemple, écrites en se référant au schéma http://schemas.org/City.



Name	Latitude	Longitude
Vancouver	49.25	-123.1
Madrid	40.4	-3.68333
Москва	55.7517	37.6178

Figure 8 : Mashup sémantique à partir de la base de connaissance de WikiNEXT

4 Usages et évaluation de WikiNEXT

En développement depuis trois ans, WikiNEXT a été développé principalement par une personne dans le cadre de sa thèse, mais la nature ouverte du wiki fait qu'il a été testé par un groupe de 5 à 6 personnes au cours de son développement. La plupart des exemples en ligne ont été développés par ce noyau de béta testeurs. Il y a quelques mois l'outil est devenu stable et nous avons procédé à des évaluations auprès de personnes n'ayant pas participé à son développement. Nous avons défini un protocole de test consistant à effectuer divers tutoriaux, notamment l'application décrite dans la section 4. Le groupe de test était constitué de développeurs web et d'étudiants en master ou en thèse d'informatique. Nous avons mesuré le temps passé pour chaque étape, observé les comportements et noté tous les problèmes rencontrés. Finalement nous avons conduit des interviews avec chaque personne et demandé de comparer cette expérience avec un développement classique pour effectuer les mêmes tâches. Nos web développeurs ont suivi un cours sur le web sémantique et ont une petite expérience de développement dans ce domaine. Les résultats[23] ont montré que WikiNEXT est jugé simple à utiliser et propose une manière originale et rapide pour requêter le web de données et réutiliser les données sémantiques. La principale difficulté rencontrée par les utilisateurs a été l'écriture de requêtes SPARQL : la syntaxe en elle-même a parfois posé problème, notamment la nécessité d'encadrer certains caractères avec des anti slash en JavaScript, mais aussi l'apprentissage de vocabulaires nouveaux. Nous avons depuis écrit des outils d'aide au requêtage (traitement automatique des caractères spéciaux pour JavaScript, widget avec des requêtes prêtes à l'emploi, historique des requêtes récentes, auto-complétion sur les propriétés des vocabulaires, etc.). Ces applications[24] d'aide ont toutes été écrites dans WikiNEXT, ce qui montre son potentiel.

Des expérimentations à plus grande échelle, nécessaires, sont en cours auprès d'étudiants qui suivent le cours Web Sémantique en master 2 informatiques à l'Université de Nice, et avec les étudiants internationaux qui suivent le cours HTML5 du W3C[25]. Par ailleurs nous développons actuellement une application de synthèse des activités des équipes de l'INRIA Sophia-Antipolis en exploitant le rapport d'activité de l'année 2012 disponible dans une version RDF/JSON. Une telle application a déjà été développée à l'aide de Semantic Media Wiki, ce qui constituera une bonne base de comparaison. Les auteurs de cette application sont impliqués dans la nouvelle expérimentation.

5 Discussion et conclusion

WikiNEXT propose une approche originale en mélangeant wiki sémantique et environnement de développement basé web (« web-based IDE »). S'appuyant sur des technologies émergentes, il propose aujourd'hui un environnement intégré pour programmer le web de données. Ses applications sont principalement pédagogiques (pour enseigner le web de données ou HTML5) mais aussi, il concurrence directement les outils de développement basés web comme jsbin.com ou jsfiddle.net, en ajoutant un moteur de templates, une base de connaissances, une puissante API pour manipuler des données sémantiques et pour requêter le web de données. Des tests d'usage ont été menés auprès de développeurs web et d'étudiants de master informatique, avec succès. En ajoutant l'aspect « programmation d'application dans la navigateur » il étend le concept de wiki sémantique en ouvrant de nouveaux champs d'application. L'outil est en ligne et propose plusieurs tutoriaux. Nous poursuivons son développement et allons procéder cette année à de nouvelles évaluations.

[23] Le protocole de test, les résultats, le questionnaire, sont disponibles : http://wikinext.gexsoft.com/wiki/52304d287a61be9c29000011

[24] Voir par exemple l'outil d'aide au requêtage SPARQL : http://wikinext.gexsoft.com/wiki/51d540a7528188fb3c000038

[25] Le cours HTML5 du W3C est disponible sur http://w3devcampus.com. Il a été écrit par un des auteurs de cet article. WikiNEXT a été utilisé pour écrire certains exemples interactifs et tutoriaux.

Références

AUER, S., DIETZOLD, S., & RIECHERT, T. (2006). OntoWiki–A tool for social, semantic collaboration. In The Semantic Web-ISWC 2006 (pp. 736-749). Springer Berlin Heidelberg.

ARAPOV, P., & BUFFA, M. (2012, April). WikiNext, a JavaScript semantic wiki. In developer track, WWW2012 Conference, Lyon.

ARAPOV, P., & BUFFA, M. (2012, August). WikiNext, a JavaScript wiki with semantic features. In Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration (p. 39). ACM.

BUFFA, M. (2005) Michel Buffa "Intranet Wikis", workshop "intrawebs" of WWW 2006, Edinburgh, Scotland.

BUFFA, GANDON, ERETEO, SANDER, & FARON (2008), SweetWiki: A semantic wiki, Special Issue of the Journal of Web Semantics on Semantic Web and Web 2.0, Volume 6, Issue 1, February 2008 , Edited by Mark Greaves and Peter Mika, Elsevier, Pages 84-97

BUFFA, M., HUSSON, G. (2012) : « SweetDeki : le wiki sémantique couteau suisse du réseau social ISICIL », 12ème Conférence Internationale Francophone "Extraction et Gestion des Connaissances" EGC'12, Janvier 2012, Bordeaux, pages 387-399

CRESPO, ARTURO & BIER, ERIC A. (1996) WebWriter: A browser-based editor for constructing web applications. Computer Networks and ISDN Systems, 1996, vol. 28, no 7, p. 1291-1306.

DI IORIO, A., MUSETTI, A., PERONI, S., VITALI, F. (2010) : Ontology-driven generation of wiki content and interfaces. The New Review of Hypermedia and Multimedia 16(1&2): 9-31 (2010)

OREN E. SemperWiki: a semantic personal Wiki. In SemDesk. 2005

GANDON, ABDESSALEM, BUFFA, & AL. (2009), ISICIL: Information Semantic Integration through Communities of Intelligence online, 10th IFIP Working Conference on Virtual Enterprises, Thessaloniki, Greece, 7-9 October 2009.

KRÖTZSCH. M., VRANDEČIĆ. D.. & VÖLKEL. M. (2006). Semantic mediawiki. In The Semantic Web-ISWC 2006 (pp. 935-942). Springer Berlin Heidelberg.

KROTZSCH, MARKUS, SCHAFFERT, SEBASTIAN, & VRANDECIC (2007), DENNY. Reasoning in semantic wikis. In: Reasoning Web. Springer Berlin Heidelberg, 2007. p. 310-329.

HERNANDEZ A. G. AND GARCIA M. N. M. A JavaScript RDF store and application library for linked data client applications. In Devtracks of the, WWW2012, conference. Lyon, France. 2012

LEUF, B., & CUNNINGHAM, W. (2001). The Wiki way: quick collaboration on the Web, Book.

MEILENDERA, T., JAYB, N., LIEBERB, J., & PALOMARESA, F (2010). Semantic wiki engines: a state of the art. Semantic-web-journal.net. IOS Press, 2010.

OREN, E., BRESLIN, J. G., & DECKER, S. (2006, May). How semantics make better wikis. In Proceedings of the 15th international conference on World Wide Web (pp. 1071-1072). ACM

RECH, J., BOGNER, C., & HAAS, V. (2007). Using wikis to tackle reuse in software projects. Software, IEEE, 24(6), 99-104.

ROSPOCHER, M., GHIDINI, C., PAMMER, V., SERAFINI, L., LINDSTAEDT, S.: (2009) MoKi: the Modelling wiKi. SemWiki 2009

SCHAFFERT, S. (2006, June). IkeWiki: A semantic wiki for collaborative knowledge management. In Enabling Technologies: Infrastructure for Collaborative Enterprises, 2006. WETICE'06. 15th IEEE International Workshops on (pp. 388-396). IEEE.

SCHAFFERT, S., EDER, J., SAMWALD, M. & BLUMAUER, A. (2008). Kiwi - knowledge in a wiki. In European Semantic Web Conference 2008.